```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSS
```

```
_$

Sy
---
NT
NT
NT
NT
NT
NT



NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT
NT

NT

NT
NT
NT
NT
NT
NT
NT



NT
NT
NT
NT
NT
NT
PI
```

K 15

```
**FILE**ID**RMOJOURNL
```

```
RRRRRRRR    MM      MM    000000            JJ    000000    UU      UU  RRRRRRRR    NN      NN  LL
RRRRRRRR    MM      MM    000000            JJ    000000    UU      UU  RRRRRRRR    NN      NN  LL
RR      RR  MMMM  MMMM  00      00          JJ  00      00  UU      UU  RR      RR  NN      NN  LL
RR      RR  MMMM  MMMM  00      00          JJ  00      00  UU      UU  RR      RR  NN      NN  LL
RR      RR  MM  MM  MM  00    0000          JJ  00      00  UU      UU  RR      RR  NNNN    NN  LL
RR      RR  MM  MM  MM  00    0000          JJ  00      00  UU      UU  RR      RR  NNNN    NN  LL
RRRRRRRR    MM      MM  00  00  00          JJ  00      00  UU      UU  RRRRRRRR    NN  NN  NN  LL
RRRRRRRR    MM      MM  00  00  00          JJ  00      00  UU      UU  RRRRRRRR    NN  NN  NN  LL
RR  RR      MM      MM  0000    00  JJ      JJ  00      00  UU      UU  RR  RR      NN    NNNN  LL
RR  RR      MM      MM  0000    00  JJ      JJ  00      00  UU      UU  RR  RR      NN    NNNN  LL
RR    RR    MM      MM  00      00  JJ      JJ  00      00  UU      UU  RR    RR    NN      NN  LL              ....
RR    RR    MM      MM  00      00  JJ      JJ  00      00  UU      UU  RR    RR    NN      NN  LL              ....
RR      RR  MM      MM    000000      JJJJJJ      000000    UUUUUUUUUU  RR      RR  NN      NN  LLLLLLLLLL      ....
RR      RR  MM      MM    000000      JJJJJJ      000000    UUUUUUUUUU  RR      RR  NN      NN  LLLLLLLLLL      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

RMO
VO4

```
0000          $BEGIN  RMOJOURNL,000,RM$RMS_JOURNAL,<RMS Journaling Manager>
0000
0000    ;
0000    ;*****************************************************************************
0000    ;*                                                                           *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000    ;*  ALL RIGHTS RESERVED.                                                     *
0000    ;*                                                                           *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    ;*  TRANSFERRED.                                                             *
0000    ;*                                                                           *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    ;*  CORPORATION.                                                             *
0000    ;*                                                                           *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    ;*                                                                           *
0000    ;*                                                                           *
0000    ;*****************************************************************************
0000    ;
0000    ;++
0000    ; Facility:       RMS-32
0000    ;
0000    ; Abstract:
0000    ;                 This module provides an interface between RMS and the
0000    ;                 Common Journaling Facility.
0000    ;
0000    ; Environment:
0000    ;                 VAX/VMS Operating System
0000    ;
0000    ; Author:         Jeffrey W. Horn,                Creation Date:  17-Mar-1982
0000    ;
0000    ; Modified By:
0000    ;
0000    ;         V03-044 JWT0162         Jim Teague                8-Mar-1984
0000    ;                 Disable RM$RTVJNL for now.
0000    ;
0000    ;         V03-043 JWT0160         Jim Teague                29-Feb-1984
0000    ;                 Remove calls to RM$DEALLEFN.
0000    ;
0000    ;         V03-042 DAS0014         David Solomon             08-Feb-1984
0000    ;                 Specify ACE$M_NOPROPAGATE for RMSJNLID ACE (they should never
0000    ;                 be propagated, as they are meaningful to only one file). Fix bug
0000    ;                 that journal name ACEs were not being marked hidden/protected.
0000    ;
0000    ;         V03-041 DAS0013         David Solomon             21-Dec-1983
0000    ;                 Support BRO access for journaling.
0000    ;
0000    ;         V03-040 JWT0141         Jim Teague                11-Nov-1983
0000    ;                 Change IFB$V_RUM to IFB$V_ONLY_RU
0000    ;
```

```
0000   ;       V03-039 KPL0015         Peter Lieberwirth       27-Oct-1983
0000   ;               Fix bug introduced in V03-038.  Symptom was breaking relative
0000   ;               file extend journaling.
0000   ;
0000   ;       V03-038 KPL0014         Peter Lieberwirth       20-Oct-1983
0000   ;               If doing AI or BI recovery, avoid allocating IRAB JNLBDB
0000   ;               and buffer in CONJNL.  This is due to interactions with
0000   ;               setting IFB BIO and a recovery process being the only type
0000   ;               of process permitted to journal a file open for mixed
0000   ;               block and record access (BRO).  Symptom is an FTL$_DEALLER
0000   ;               bugcheck because a JNLBDB gets allocated and dropped when
0000   ;               another is allocated in RM$WRITE.  (Bugcheck happens on
0000   ;               close.)
0000   ;
0000   ;       V03-037 KPL0013         Peter Lieberwirth       11-Oct-1983
0000   ;               Deallocate EFNs after finishing with them.  Improper use
0000   ;               of EFNs is causing hangs in asynch situations.  Fix problem
0000   ;               with non-page aligned ALDJNLBUF allocations.
0000   ;
0000   ;       V03-036 DAS0012         David Solomon           27-Sep-1983
0000   ;               Preserve R3 in RM$WRTJNL (ISAM assumed it was preserved).
0000   ;               Corrected some comments.
0000   ;
0000   ;       V03-035 DAS0011         David Solomon           08-Sep-1983
0000   ;               Correct overzealous fix to RM$DSCJNL in V03-034. Fix test in
0000   ;               RM$MAPJNL that decides whether or not this is an open entry.
0000   ;               Return RMS$_JNF if no journal name specified, vs RMS$_NOJ.
0000   ;
0000   ;       V03-034 DAS0010         David Solomon           25-Aug-1983
0000   ;               Fix accvio when no journal name is specified. Set up R10 before
0000   ;               call to RM$RETJNLBDB (also caused an accvio). Use correct ACE
0000   ;               field name for RMS journal names. Replace source.
0000   ;
0000   ;       V03-033 LJA0090         Laurie J. Anderson      18-Aug-1983
0000   ;               1) Fix the writing of the journal entries to not stuff in
0000   ;                  the version number as VER1 but rather as the constant
0000   ;                  MAXVER so that when the versions are increased (as I
0000   ;                  just did) the new version number is filled in.
0000   ;               2) Fill in a new (RJR version V04-000 field - for AT journals
0000   ;                  the FAB/RAB user CTX field, so that it is written to
0000   ;                  the journal for the users discretion.
0000   ;               3) Now that the FAB is available when filling in the RJR
0000   ;                  use the completion status from it, rather than just
0000   ;                  stuff success.
0000   ;
0000   ;       V03-032 KPL0012         Peter Lieberwirth       30-Jul-1983
0000   ;               Allocate a bigger JNLBDB Buffer id AI journaling a relative
0000   ;               file.  The larger buffer will be used for the prolog if
0000   ;               the file is created.
0000   ;
0000   ;       V03-031 KPL0011         Peter Lieberwirth       24-Jul-1983
0000   ;               Fill in file-oriented AT journal record during MAPJNL
0000   ;               call.  Data from IFAB is used to fill in some create/open/close
0000   ;               AT fields.  RM$AT_JOURNAL_RECORD fills in some RJR RAB data.
0000   ;               RM$AT_COM_RAB added to fill AT record in with initial user
0000   ;               search and operation input.
0000   ;
```

```
0000  ;                 Also, fix error paths and block-IO success status path in
0000  ;                 RM$CONJNL.
0000  ;
0000  ;                 Also, use RM$ALDJNLBUF and RM$RETJNLBDB to allocate and
0000  ;                 deallocate journaling-specific BDB/Buffers.  Can't just use
0000  ;                 ALDBUF etc... because then the BDB will be linked into the
0000  ;                 IFABs BDB list - and could get used for file IO.  Also,
0000  ;                 now the file-related AT BDB/Buffer can remain allocated for
0000  ;                 the duration of the file open - previously it was deallocated
0000  ;                 at common create/open exit because all BDBs on the IFAB
0000  ;                 list were deallocated at that time.
0000  ;
0000  ;                 Add some commentary about RMS Journaling
0000  ;
0000  ;         V03-030 KPL0010         Peter Lieberwirth       1-Jul-1983
0000  ;                 Fix FORCE_JNL to always return status.
0000  ;
0000  ;         V03-029 KPL0009         Peter Lieberwirth       16-Jun-1983
0000  ;                 Fix some bugs.  Add routine to write AT journal records for
0000  ;                 record operations.  Clean up RM$MAPJNL to let it write AT
0000  ;                 file operation records.  Remove COP and CQE in favor of CJF.
0000  ;                 Move misc IFAB jnl flags to JNLFLG2.
0000  ;
0000  ;         V03-028 TSK0052         Tamar Krichevsky        5-jun-1983
0000  ;                 Fix bugs introduced by V03-26.  Move module to RM$RMS_JOURNAL
0000  ;                 psect.  Fix broken branches to RM$MAPERR.
0000  ;
0000  ;         V03-027 KPL0008         Peter Lieberwirth       30-May-1983
0000  ;                 Fix bugs introduced in V03-026 and earlier.
0000  ;
0000  ;         V03-026 KPL0007         Peter Lieberwirth       26-May-1983
0000  ;                 Support new more robust RJR format.  Fix typos in KPL0001.
0000  ;                 Turn on sequential file journaling.  Rework RJB/BDB allocation.
0000  ;
0000  ;         V03-025 TSK0050         Tamar Krichevsky        25-May-1983
0000  ;                 Modify RM$CONJNL to allocate the proper size journal buffer
0000  ;                 for sequential files.  Currently, the user specified bucket
0000  ;                 size is used to determine the buffer's length.  For sequential
0000  ;                 files, the buffer must be large enough to contain any one
0000  ;                 record from the file.
0000  ;                 Cleanup calculation of overhead for journal buffer.
0000  ;
0000  ;         V03-024 DAS0009         David Solomon           11-May-1983
0000  ;                 Fix WRTACC check in RM$ASSJNL (BBC to BBS). Add missing "#"
0000  ;                 in front of two literals that were causing accvio's.  Fix
0000  ;                 error path on failure to assign channel to RU journal.  Clear
0000  ;                 pointer to RJB upon its deallocation.  Don't allocate IRAB
0000  ;                 AT journal buffer if not AT journaling.  Fix ALLOC_MJB to
0000  ;                 acquire space from same page as IFAB.  Do better job at
0000  ;                 calculating required size of MJB.
0000  ;
0000  ;         V03-023 KPL0006         Peter Lieberwirth       2-May-1983
0000  ;                 Turn on $WRITEJNL call.  Add $WRMODDEF.  Fix bug on
0000  ;                 error path into RM$DEAJNL.
0000  ;
0000  ;         V03-022 KPL0005         Peter Lieberwirth       1-May-1983
0000  ;                 Delete obsolete MJB definitions.
```

```
0000  ;
0000  ;       V03-021 KPL0004         Peter Lieberwirth       1-May-1983
0000  ;               Fix another problem with $WRITEJNL call.
0000  ;
0000  ;       V03-020 KPL0003         Peter Lieberwirth       1-May-1983
0000  ;               Fix call to $WRITEJNL.
0000  ;
0000  ;       V03-019 KPL0002         Peter Lieberwirth       30-Apr-1983
0000  ;               Add omitted macro definition.  Flesh out WRITE_MJB
0000  ;               routine.
0000  ;
0000  ;       V03-018 KPL0001         Peter Lieberwirth       29-Apr-1983
0000  ;               Allocate miscellaneous journaling buffers for IFB and IRB
0000  ;               where necessary.  Generalize cleanup so these always get
0000  ;               deallocated.  Add stub RM$WRITE_MJB routine.
0000  ;
0000  ;       V03-017 JWH0221         Jeffrey W. Horn         26-Apr-1983
0000  ;               If in recovery allow BRO access.  Also temporarily, enable
0000  ;               both AI and BI journaling durring recovery.
0000  ;
0000  ;       V03-016 JWH0205         Jeffrey W. Horn         11-Apr-1983
0000  ;               Implement journal id ACE.  Also add protected and hidden
0000  ;               bits to all ACEs.
0000  ;
0000  ;       V03-015 DAS0008         David Solomon           01-Apr-1983
0000  ;               Save R2 in RM$WRTJNL (for ISAM).
0000  ;
0000  ;       V03-014 RAS0135         Ron Schaefer            17-Mar-1983
0000  ;               More corrections to RAS0132 for registers and RJR$_ names.
0000  ;
0000  ;       V03-013 RAS0135         Ron Schaefer            17-Mar-1983
0000  ;               Corrections to RAS0132 for registers and RJR$_ names.
0000  ;
0000  ;       V03-012 RAS0132         Ron Schaefer            16-Mar-1983
0000  ;               Merge $RMSRDEF into $RJRDEF and revise the interface
0000  ;               for RM$WRTJNL for easier use from ISAM.
0000  ;
0000  ;       V03-011 JWH0185         Jeffrey W. Horn         11-Feb-1983
0000  ;               Set WRFLG$V_BI on RU journal entries.
0000  ;               Use the perm FWA to provide journal entry security and
0000  ;               to fill in the mapping entries.
0000  ;               If file is opened UFO then disable journaling for this open.
0000  ;
0000  ;       V03-010 JWH0180         Jeffrey W. Horn         03-Feb-1983
0000  ;               Change references to RJR$C_MAPLEN from byte to word.
0000  ;
0000  ;       V03-009 JWH0173         Jeffrey W. Horn         24-Jan-1983
0000  ;               Clean up status code returns.
0000  ;               Use BKS instead of MRS to allocate journal BDB.
0000  ;               Allow ISAM journaling.
0000  ;
0000  ;       V03-008 JWH0167         Jeffrey W. Horn         10-Jan-1983
0000  ;               Implement IFB recovery option byte.
0000  ;               Fill in file organization in mapping entry.
0000  ;
0000  ;       V03-007 JWH0155         Jeffrey W. Horn         3-Dec-1982
0000  ;               Seperate journal names into three seperate ACEs.
```

```
0000   ;               Prevent journaling on Sequential and Indexed files.
0000   ;               For block io, do not create journal BDB and buffer.
0000   ;
0000   ;       V03-006 JWH0154         Jeffrey W. Horn         13-Dec-1982
0000   ;               Define ACE$C_JNLNAMS (temporary).
0000   ;
0000   ;       V03-005 JWH0132         Jeffrey W. Horn         22-Nov-1982
0000   ;               Write journal entries with the WRFLG$M_LOCK attribute.
0000   ;
0000   ;       V03-004 JWH0128         Jeffrey W. Horn         15-Nov-1982
0000   ;               Change SS$_NOCJF code to SS$_IVSSRQ.
0000   ;
0000   ;       V03-003 JWH0116         Jeffrey W. Horn         28-Oct-1982
0000   ;               If in RCP then don't perfom any journaling execpt AT.
0000   ;               Remove CALLS to CJF services and replace with macros.
0000   ;               Change logic in FRCJNL which checks for an active RU to
0000   ;               reflect changes in RUF.
0000   ;
0000   ;       V03-002 JWH0108         Jeffrey W. Horn         23-Sep-1982
0000   ;               Remove redefinitions of ACL ACP attributes.
0000   ;               Fix problem with setting size for RJB deallocation.
0000   ;               Clean up status code returns.
0000   ;               Redefine journal names (FWA$T_xxJNLN) as .ASCIC
0000   ;               strings.
0000   ;               Implement new RMS journaling record (RJR).
0000   ;               Use RM$GETBLK and RM$RETBLK instead of RM$GETSPC and
0000   ;               RM$RETSPC when allocating and deallocting the RJB.
0000   ;
0000   ;       V03-001 JWH0107         Jeffrey W. Horn         23-Sep-1982
0000   ;               Redefine ACL ACP attributes to ATR$C_USERLABEL which is a
0000   ;               no-op.  Add a .WEAK for CJF$GETJNL.  Clean up status code
0000   ;               returns.
0000   ;
0000   ;--
```

```
          0000          .SBTTL  DECLARATIONS
          0000
          0000
          0000  ;
          0000  ; Include Files:
          0000  ;
          0000
          0000  ;
          0000  ; Macros:
          0000  ;
          0000
          0000          $ACEDEF
          0000          $ATRDEF
          0000          $BDBDEF
          0000          $CJFDEF
          0000          $DVIDEF
          0000          $FABDEF
          0000          $RABDEF
          0000          $FIBDEF
          0000          $FWADEF
          0000          $IFBDEF
          0000          $IODEF
          0000          $IMPDEF
          0000          $IRBDEF
          0000          $PCBDEF
          0000          $PSLDEF
          0000          $RJBDEF
          0000          $RJRDEF
          0000          $RMSDEF
          0000          $RUCBDEF
          0000          $SSDEF
          0000          $STSDEF
          0000          $WRFLGDEF
          0000          $MJBDEF
          0000          $WRMODDEF
          0000
          0000  ;
          0000  ; Equated Symbols:
          0000  ;
          0000
          0000  ;
          0000  ; Own Storage:
          0000  ;
          0000
    0001  0000  FACILITY:       .WORD   RMS$_FACILITY
    0001  0002  MODE:           .WORD   PSL$C_EXEC
```

```
0004          .SUBTITLE Introduction to RMS Journaling
0004   ;++
0004   ;      RMS Journaling Manager
0004   ;
0004   ; This module contains routines used to journal RMS operations.  Other modules
0004   ; containing journaling routines (not necessarily an inclusive list) are:
0004   ;
0004   ;      RM3JOURNL.B32, RM1JOURNL.MAR, RMOCRECOM.MAR, RMOBUFMGR.MAR,
0004   ;      RMOEXTEND.MAR, and RM2CREATE.MAR
0004   ;
0004   ; The data structures are defined in:
0004   ;
0004   ;      RMSINTSTR.MDL and the format of the RMS Journaling Record (RJR) is
0004   ; described in RMSFILSTR.SDL.
0004   ;
0004   ; The general flow of journaling control is as follows:
0004   ;
0004   ; 1. When a file marked for journaling is accessed, connections are made
0004   ;    to the journals specified in the file's header in RM$ASSJNL.  Certain
0004   ;    data structures are allocated at this time also.
0004   ;
0004   ; 1a. If the file is being created, the data structures are allocated earlier,
0004   ;    and the JNLXAB is interrogated for journal names.  If no journal names
0004   ;    are specified in the XAB, CJF is asked for default journal names.  This
0004   ;    is done in RM$GETJNL.
0004   ;
0004   ; 2. RM$MAPJNL is called to write entries to the journals at OPEN/CREATE/CLOSE
0004   ;    time.  These entries contain the full filename and other information.
0004   ;    These entries are used when the journal must be interrogated for file
0004   ;    names, and to associate a filename with a journal ID.
0004   ;
0004   ;      A journal ID is a unique identifier associated with a journaled file
0004   ;      (it is kept in the file header in a hidden, protected, access control
0004   ;      entry).  It is used in most RMS journaling records so that the full
0004   ;      filename need not be kept in all entries.  It is also used as a
0004   ;      short-hand identifier to search a journal for RMS entries without
0004   ;      having to fully specify the filename as originally journaled.
0004   ;
0004   ;
0004   ; 3. RM$CONJNL is called at connect time to allocate record-oriented RMS
0004   ;    journaling structures.  These include buffers and buffer descriptors.
0004   ;    These structures are deallocated at disconnect time in RM$DSCJNL.
0004   ;    RM$DSCJNL also forces to the journal any audit-trail journal entries
0004   ;    written to CJF but not yet necessarily forced to the actual journal
0004   ;    (IE the entries may still be in a CJF buffer.)
0004   ;
0004   ; 4. During the course of RMS record operations journal entries describing
0004   ;    file accesses and modifications are written to the appropriate journals.
0004   ;
0004   ;      ISAM AI and BI operations are journaled by writing copies of the
0004   ;      modified buckets to the journal.  The buffers used for these entries
0004   ;      are as follows:
0004   ;
0004   ;              AI - the buffer used is the actual data bucket that is written
0004   ;                   to the file
0004   ;
0004   ;              BI - the buffer used is an extra one allocated at the same time
```

```
0004  ;                      the data buffer is allocated
0004  ;
0004  ;                 Both buffers are pointed to by the BDB.
0004  ;
0004  ;         ISAM AI and BI operations are journaled at the bucket-level because
0004  ;         there was no way found to journal on a record basis and ensure that
0004  ;         RFAs would be restored upon recovery.
0004  ;
0004  ;         ISAM recovery unit operations are journaled by writing information
0004  ;         describing the modified record to the journal.  The ISAM code treats
0004  ;         record operations in recovery units in a special fashion:
0004  ;
0004  ;                 $DELETEs do not delete the record - the record is merely
0004  ;                 marked for deletion.
0004  ;
0004  ;                 $UPDATEs never shrink the size of the record - extra space
0004  ;                 corresponding to the original size of the record is kept
0004  ;                 and described by special fields in the record itself.
0004  ;
0004  ;                 The reason for never deleting space in ISAM RUs is to ensure
0004  ;                 there will always be space in the bucket if the record
0004  ;                 must be rolled back in.  We don't want to invent more
0004  ;                 special case ISAM bucket split code.  The RFA basis of the
0004  ;                 journal entry also precludes too much bucket entropy before
0004  ;                 recovery.
0004  ;
0004  ;         Sequential and Relative file journaling is done on a record basis.
0004  ;         A record journaling buffer is allocated at CONNECT time, and this
0004  ;         buffer is used to build the record used to describe the change needed
0004  ;         to undo or redo the operation.
0004  ;
0004  ;         Audit-trail journaling is done on a file and record level.  A special
0004  ;         BDB and Buffer is allocated off the IFAB to contain file related
0004  ;         audit-trail information.  A journaling buffer descriptor/buffer
0004  ;         is allocated off the IRAB to collect and format record-related
0004  ;         audit trail information.
0004  ;
0004  ;         In order to ensure ISAM AI recovery, $EXTENDs must be journaled.
0004  ;         A special extend buffer descriptor/buffer is allocated off the
0004  ;         IFAB - the journaling record to describe the extend is built in
0004  ;         and written from this buffer.  Sequential and Relative AI extends
0004  ;         are journaled in the same fashion.
0004  ;
0004  ; 5. RMS Journaling Data Structures
0004  ;
0004  ;         RJB     - The RJB is allocated by ASSJNL or CRECOM, and contains
0004  ;                 the channels assigned to various journals.  Flags indicating
0004  ;                 connections to journals are also present.
0004  ;
0004  ;         IFB JNLFLG - This byte is a copy of the file header byte which
0004  ;                 indiates what types of journaling the file is marked for.
0004  ;
0004  ;         IFB JNLFLG2 - This byte contains miscellaneous run-time IFAB related
0004  ;                 journaling indicators.
0004  ;
0004  ;         IFB$L_JNLBDB - This field points to a BDB and buffer that is used for
0004  ;                 file related AT journaling.
```

```
0004  ;
0004  ;       IFB$L_ATJNLBUF - This field points into the buffer pointed to indirectly
0004  ;               by IFB$L_JNLBDB.  This field points directly to the RJR within
0004  ;               the buffer.
0004  ;
0004  ;       RJR - RMS Journaling Record.  The format of the RMS data written to
0004  ;               the journal.  It is comprised of a common overhead, and several
0004  ;               different formats following the common overhead that are used
0004  ;               for different journaling functions.
0004  ;
0004  ;               Currently implemented:  FILE, RECORD, BLOCK, BUCKET, EXTEND,
0004  ;               AT_RECORD.
0004  ;
0004  ;       MJB - Miscellaneous Journaling Block  This is used to describe
0004  ;               miscellaneous journaling records and the information needed
0004  ;               to describe the WRITEJNL request.  The MJB is written by
0004  ;               RM$WRITE_MJB and is forced to the journal by RM$FORCE_MJB.
0004  ;
0004  ;               MJBs are currently used for AT and Extend entries.
0004  ;
0004  ;       IRB$L_ATJNLBUF - points to an MJB/Buffer used to write record level AT
0004  ;               entries.
0004  ;
0004  ;       Why MJBs and BDBs?  Good question.  The BDB related design is good for
0004  ;       writing buffers containing actual file data to the journals.  The
0004  ;       MJB is used when descriptive entries not directly related to file
0004  ;       data are written.  BDB/Buffer fits into the IO system concept and
0004  ;       ISAM AI and BI benefits from the overlap.  MJB/Buffer fits into
0004  ;       the CJF design better.  The MJB describes the WRITEJNL inputs,
0004  ;       basically.  The only counter-intuitive setup currently is writing
0004  ;       file-level descriptive entries via BDB and not MJB.  The reason for this
0004  ;       is that MAPJNL was originally set up this way.
0004  ;
0004  ;--
```

```
                            0004         .SBTTL  RM$GETJNL - Get Journal Name
                            0004
                            0004  ;++
                            0004  ; RM$GETJNL - Get Journal Name
                            0004  ;
                            0004  ;       This subroutines gets the journal names to use from either CJF
                            0004  ;       or the process-based default journal names.  It then proceeds to
                            0004  ;       set up the attributes for the file creation.
                            0004  ;
                            0004  ;
                            0004  ; Calling sequence:
                            0004  ;
                            0004  ;       BSBW    RM$GETJNL
                            0004  ;
                            0004  ; Input Parameters:
                            0004  ;
                            0004  ;       R9      -       IFAB address
                            0004  ;       R10     -       FWA address
                            0004  ;
                            0004  ; Implicit Inputs:
                            0004  ;
                            0004  ;       IFB$B_JNLFLG -  File's Journaling Flags
                            0004  ;       FWA$L_UIC    -  File's Owner UIC
                            0004  ;       FWA$Q_xxJNL, FWA$T_xxJNLN - may be preset by XAB processing to contain
                            0004  ;                                   some journal names.
                            0004  ;
                            0004  ; Output Parameters:
                            0004  ;
                            0004  ;       R1-R4           Destroyed
                            0004  ;
                            0004  ; Implicit Outputs:
                            0004  ;
                            0004  ;       FWA$Q_xxJNL, FWA$Q_xxJNLN - Set to journal name(s).
                            0004  ;
                            0004  ; Completion Codes:
                            0004  ;
                            0004  ;       JNF - If no journal name found for a particular IFB$B_JNLFLG bit,
                            0004  ;               STV will contain CJF status from $GETJNL.
                            0004  ;
                            0004  ; Side Effects:
                            0004  ;       None.
                            0004  ;
                            0004  ;--
                            0004
                            0004  RM$GETJNL::
         7E    01    D0     0004          MOVL    #1,-(SP)                              ; anticipate success
16 00A0  C9    02    E1     0007          BBC     #IFB$V_BI,IFB$B_JNLFLG(R9),10$        ; branch if no BI bit
   52  08C8    CA    9E     000D          MOVAB   FWA$Q_BIJNL(R10),R2                   ; fwa bi descr
   53  08E0    CA    9E     0012          MOVAB   FWA$T_BIACE(R10),R3                   ; fwa bi buffer
         54    02    D0     0017          MOVL    #CJF$_BI,R4                           ; journal type code
             0084    30     001A          BSBW    GET_JNL                               ; get journal name
          03   50    E8     001D          BLBS    R0,10$                                ; get out on error
         6E    50    D0     0020          MOVL    R0,(SP)                               ; remember error code
                            0023
16 00A0  C9    03    E1     0023  10$:    BBC     #IFB$V_AI,IFB$B_JNLFLG(R9),20$        ; branch if no AI bit
   52  08D0    CA    9E     0029          MOVAB   FWA$Q_AIJNL(R10),R2                   ; fwa AI descr
   53  08F4    CA    9E     002E          MOVAB   FWA$T_AIACE(R10),R3                   ; fwa AI buffer
```

```
            54   03  D0 0033          MOVL    #CJF$_AI,R4                                        ; journal type code
               0068  30 0036          BSBW    GET_JNL                                            ; get journal name
             03  50  E8 0039          BLBS    R0,20$                                             ; get out on error
            6E   50  D0 003C          MOVL    R0,(SP)                                            ; remember error code
                        003F
     16 00A0 C9  04  E1 003F    20$:  BBC     #IFB$V_AT,IFB$B_JNLFLG(R9),30$                     ; branch if no AT bit
         52 08D8 CA  9E 0045          MOVAB   FWA$Q_ATJNL(R10),R2                                ; fwa AT descr
         53 0908 CA  9E 004A          MOVAB   FWA$T_ATACE(R10),R3                                ; fwa AT buffer
            54   04  D0 004F          MOVL    #CJF$_AT,R4                                        ; journal type code
               004C  30 0052          BSBW    GET_JNL                                            ; get journal name
             03  50  E8 0055          BLBS    R0,30$                                             ; continue on success
            6E   50  D0 0058          MOVL    R0,(SP)                                            ; remember error code
                        005B
   092C CA  01F8 CA  D0 005B    30$:  MOVL    <FWA$T_FIBBUF+FIB$W_FID>(R10),FWA$T_FID(R10) ; put fid in id ace
   0930 CA  01FC CA  B0 0062          MOVW    <FWA$T_FIBBUF+FIB$W_FID+4>(R10),<FWA$T_FID+4>(R10)
                        0069          $GETTIM_S TIMADR=FWA$Q_ID_DATE(R10)                        ; get current time
 091C CA  0E000820 8F  D0 0074          MOVL    #<<<ACE$M_PROTECTED + ACE$M_HIDDEN + ACE$M_NOPROPAGATE> -
                        007D                  @ <ACE$W_FLAGS*8>> + -
                        007D                    <ACE$C_JNLID @ <ACE$B_TYPE*8>> + -
                        007D                    FWA$S_IDACE>, FWA$T_IDACE(R10)
            85   20  B0 007D          MOVW    #FWA$S_IDACE,(R5)+                                 ; set attribute len
            85   1F  B0 0080          MOVW    #ATR$C_ADDACLENT,(R5)+                             ; set attribute type
         85 091C CA  DE 0083          MOVAL   FWA$T_IDACE(R10),(R5)+                             ; set attribute address
                        0088          RMSSUC
                        008B
            50   8E  D0 008B    50$:  MOVL    (SP)+,R0                                           ; get status code
             01  50  E9 008E          BLBC    R0,60$                                             ; skip if error
                     05 0091          RSB
                        0092
            00A0 C9  94 0092    60$:  CLRB    IFB$B_JNLFLG(R9)                                   ; turn off journaling
                        0096          RMSERR  JNF,RT                                             ; journal not found
         00000000'EF 17 0098          JMP     RM$MAPERR                                          ; go map the error and retur
```

```
                                00A1          .SBTTL  GET_JNL - Common Get Journal name routine
                                00A1
                                00A1  ;++
                                00A1  ; GET_JNL - Common Get Journal name routine
                                00A1  ;
                                00A1  ; If XAB processing did not get a particular journal name, then ask
                                00A1  ; CJF for one.
                                00A1  ;
                                00A1  ; Calling sequence:
                                00A1  ;
                                00A1  ;       BSBW    GET_JNL
                                00A1  ;
                                00A1  ; Input Parameters:
                                00A1  ;
                                00A1  ;       R2      -       Pointer to FWA$Q_xxJNL  (fwa journal name descriptor)
                                00A1  ;       R3      -       Pointer to FWA$T_xxJNLN (fwa journal name buffer)
                                00A1  ;       R4      -       CJF$_xx for the journal type
                                00A1  ;       R5      -       Address of first free slot at end of ACP attribute list
                                00A1  ;
                                00A1  ; Implicit Inputs:
                                00A1  ;
                                00A1  ;       FWA$L_UIC       File Ownership UIC.
                                00A1  ;       FWA$Q_DEVICE    Descriptor of Device name
                                00A1  ;       FWA$L_ATR_LIST  Atribute list for create
                                00A1  ;
                                00A1  ; Output Parameters:
                                00A1  ;       R5              New free ACP attribute list free slot.
                                00A1  ;
                                00A1  ; Implicit Outputs:
                                00A1  ;
                                00A1  ;       FWA$Q_xxJNL, FWA$T_xxJNLN - filled in
                                00A1  ;       FWA$T_ATR_LIST - May have journal name attributes added.
                                00A1  ;
                                00A1  ; Completion Codes:
                                00A1  ;       Any CJF from $GETJNL.
                                00A1  ;
                                00A1  ; Side Effects:
                                00A1  ;       None.
                                00A1  ;--
                                00A1
                                00A1  GET_JNL:
                                00A1
                                00A1  ;
                                00A1  ; If no journal name from XAB processing, ask CJF for one
                                00A1  ;
           7E    01     D0      00A1          MOVL    #1,-(SP)                                ; assume success
                 62     95      00A4          TSTB    (R2)                                    ; name length zero?
                 32     12      00A6          BNEQ    20$                                     ; no branch
           62    10     3C      00A8          MOVZWL  #FWA$S_BIJNLN,(R2)                      ; set up descriptor
     04 A2    04 A3     DE      00AB          MOVAL   ACE$T_RMSJNLNAM(R3),4(R2)
              28 AA     D5      00B0          TSTL    FWA$L_UIC(R10)                          ; file uic specified?
                 0D     12      00B3          BNEQ    10$                                     ; branch if so
  51   00000000'9F      D0      00B5          MOVL    @#CTL$GL_PCB,R1                         ; get PCB address
   28 AA   00BC C1      D0      00BC          MOVL    PCB$L_UIC(R1),FWA$L_UIC(R10)            ; get UIC from PCB
                                00C2
                                00C2  10$:    $GETJNL_S       -                               ; call CJF
                                00C2                  DEVNAM = FWA$Q_DEVICE(R10), -
```

```
                    00C2                         UIC    = FWA$L_UIC(R10), -
                    00C2                         JNLTYP = R4, -
                    00C2                         JNLNAM = (R2), -
                    00C2                         RSLLEN = (R2)
                    00D7
          6E   50   D0   00D7           MOVL    R0,(SP)                                 ; save return code
                    00DA
                    00DA     ;
                    00DA     ; Construct ACE to store journal name and add to attribute list
                    00DA     ;
                    00DA            ASSUME  ACE$C_BIJNL EQ CJF$_BI
                    00DA            ASSUME  ACE$C_AIJNL EQ <ACE$C_BIJNL + 1>
                    00DA            ASSUME  ACE$C_ATJNL EQ <ACE$C_AIJNL + 1>
                    00DA
   63   62   04   81   00DA     20$:   ADDB3   #ACE$T_RMSJNLNAM,(R2),(R3)            ; fill in ACE size
     01   A3   54   90   00DE            MOVB    R4,ACE$B_TYPE(R3)                     ; move type into ACE
       0600 8F   B0   00E2            MOVW    #ACE$M_HIDDEN!ACE$M_PROTECTED,-       ; move flags into ACE
         02 A3        00E6                    ACE$W_FLAGS(R3)
          85   63   9B   00E8            MOVZBW  (R3),(R5)+                            ; move atr len into list
          85   1F   B0   00EB            MOVW    #ATR$C_ADDACLENT,(R5)+                ; move atr type into list
          85   53   D0   00EE            MOVL    R3,(R5)+                              ; move atr addr into list
          50   8E   D0   00F1            MOVL    (SP)+,R0                              ; restore code
               05   00F4            RSB
```

```
                             00F5          .SBTTL  RMS$RTVJNL - Retrieve Journaling Info
                             00F5   ;++
                             00F5   ; RMS$RTVJNL - Retrieve Journaling Info
                             00F5   ;
                             00F5   ;      This subroutine adds the neccessary ACP attributes to retrieve
                             00F5   ;      both the journal selection bits and the journal names used for a file.
                             00F5   ;
                             00F5   ; Calling Sequence:
                             00F5   ;
                             00F5   ;      BSBW    RMS$RTVJNL
                             00F5   ;
                             00F5   ; Input Parameters
                             00F5   ;      R5      Address of End of attribute list
                             00F5   ;      R9      IFAB address
                             00F5   ;      R10     FWA Address
                             00F5   ;      R11     Impure Area Address
                             00F5   ;
                             00F5   ; Implicit Imputs:
                             00F5   ;      None.
                             00F5   ;
                             00F5   ; Ouput Parameters:
                             00F5   ;
                             00F5   ;      R1      Destroyed
                             00F5   ;      R5      Updated to new end of attribute list
                             00F5   ;
                             00F5   ; Implicit Outputs:
                             00F5   ;
                             00F5   ;      FWA ACP attribute list has attributes filled in to retrieve journaling
                             00F5   ;      bits and journal names.
                             00F5   ;
                             00F5   ; Completion Codes:
                             00F5   ;      None.
                             00F5   ;
                             00F5   ; Side Effects:
                             00F5   ;      None.
                             00F5   ;
                             00F5   ;--
                             00F5
                             00F5   RMS$RTVJNL::
                             00F5
                             00F5   ;**JNL** begin temporary code to tie off journaling
                         05  00F5          RSB
                             00F6   ;**JNL** end temporary code to tie off journaling
                             00F6
                             00F6   ;
                             00F6   ; Construct ACEs to get journal names and add ACP attribute
                             00F6   ;
  51   08E0 CA           DE  00F6          MOVAL   FWA$T_BIACE(R10),R1                                ; get start of ACE
  61   0214 8F           B0  00FB          MOVW    #<<ACE$C_BIJNL@<ACE$B_TYPE*8>>+FWA$S_BIACE>,(R1) ; move in ACE Type.
       85   14           B0  0100          MOVW    #FWA$S_BIACE,(R5)+                                 ; move atr len into list
       85   23           B0  0103          MOVW    #ATR$C_FNDACLTYP,(R5)+                             ; move atr type into list
       85   51           D0  0106          MOVL    R1,(R5)+                                           ; move atr addr into list
                             0109
  51   08F4 CA           DE  0109          MOVAL   FWA$T_AIACE(R10),R1                                ; get start of ACE
  61   0314 8F           B0  010E          MOVW    #<<ACE$C_AIJNL@<ACE$B_TYPE*8>>+FWA$S_AIACE>,(R1) ; move in ACE Type.
       85   14           B0  0113          MOVW    #FWA$S_AIACE,(R5)+                                 ; move atr len into list
       85   23           B0  0116          MOVW    #ATR$C_FNDACLTYP,(R5)+                             ; move atr type into list
```

```
            85   51   D0  0119        MOVL    R1,(R5)+                                   ; move atr addr into list
                          011C
       51  0908  CA   DE  011C        MOVAL   FWA$T_ATACE(R10),R1                        ; get start of ACE
       61  0414  8F   B0  0121        MOVW    #<<ACE$C_ATJNL@<ACE$B_TYPE*8>>+FWA$S_ATACE>,(R1) ; move in ACE Type,
            85   14   B0  0126        MOVW    #FWA$S_ATACE,(R5)+                         ; move atr len into list
            85   23   B0  0129        MOVW    #ATR$C_FNDACLTYP,(R5)+                     ; move atr type into list
            85   51   D0  012C        MOVL    R1,(R5)+                                   ; move atr addr into list
                          012F
       51  091C  CA   DE  012F        MOVAL   FWA$T_IDACE(R10),R1                        ; get start of ACE
 61  00000820   8F   D0  0134        MOVL    #<<ACE$C_JNLID@<ACE$B_TYPE*8>>+FWA$S_IDACE>,(R1) ; set up ACE
            85   20   B0  013B        MOVW    #FWA$S_IDACE,(R5)+                         ; move atr len into list
            85   23   B0  013E        MOVW    #ATR$C_FNDACLTYP,(R5)+                     ; move atr type into list
            85   51   D0  0141        MOVL    R1,(R5)+                                   ; move atr addr into list
                          0144
                          0144  ;
                          0144  ; Add journal control bit attributes to list
                          0144  ;
            85   01   B0  0144        MOVW    #1,(R5)+                                   ; move atr len into list
            85   1D   B0  0147        MOVW    #ATR$C_JOURNAL,(R5)+                       ; move atr type into list
       85  00A0  C9   9E  014A        MOVAB   IFB$B_JNLFLG(R9),(R5)+                     ; move atr addr into list
                          014F
                          014F  ;
                          014F  ; Make sure we have the file's UIC in the FWA
                          014F  ;
            85   04   B0  014F        MOVW    #4,(R5)+                                   ; move atr len into list
            85   1A   B0  0152        MOVW    #ATR$C_UIC_RO,(R5)+                        ; move atr type into list
         85   28  AA  DE  0155        MOVAL   FWA$L_UIC(R10),(R5)+                       ; move atr addr into list
                          0159
                      05  0159        RSB
```

```
                    015A          .SBTTL  RM$ASSJNL - Open Journaling for a file
                    015A
                    015A  ;++
                    015A  ; RM$ASSJNL - Open Journaling for a file
                    015A  ;
                    015A  ;       This subroutine builds the necessary data structures for journaling
                    015A  ;       onto the IFAB and opens the journals needed for the file.
                    015A  ;
                    015A  ; Calling sequence:
                    015A  ;
                    015A  ;       BSBW    RM$ASSJNL
                    015A  ;
                    015A  ; Input Parameters:
                    015A  ;
                    015A  ;       R8      FAB Address
                    015A  ;       R9      IFAB Address
                    015A  ;       R10     FWA Address
                    015A  ;       R11     Impure Area Address
                    015A  ;
                    015A  ; Implicit Inputs:
                    015A  ;
                    015A  ;       IFB$B_JNLFLG
                    015A  ;
                    015A  ; Output Parameters:
                    015A  ;
                    015A  ;       R1 - R5 Destroyed
                    015A  ;
                    015A  ; Implicit Outputs:
                    015A  ;
                    015A  ;       IFB$L_RJB       Address of allocated and initialized RJB
                    015A  ;       IFB$B_JNLFLG2   Files Journaling Flags:
                    015A  ;               IFB$V_JNL       Set to indicate journaling initialized for this
                    015A  ;                               file.
                    015A  ;
                    015A  ; Completion Codes:
                    015A  ;
                    015A  ;       Any RMS, particularly, DME.
                    015A  ;       NOJ, Journal device for file not available, CJF status in
                    015A  ;               STV from $ASSJNL.
                    015A  ;       JNS, Journaling not supported for operation
                    015A  ;
                    015A  ; Side Effects:
                    015A  ;       None.
                    015A  ;
                    015A  ;--
                    015A
                    015A  ERRJNS: RMSERR  JNS
               05   015F          RSB
                    0160
     00A0 C9   94   0160  UFO:    CLRB    IFB$B_JNLFLG(R9)                        ; turn off journaling
                    0164  ASS_DONE:
                    0164          RMSSUC
               05   0167          RSB
                    0168
                    0168  RM$ASSJNL::
F6 00A2 C9  04 E2   0168          BBSS    #IFB$V_DONE_ASS_JNL,IFB$B_JNLFLG2(R9),ASS_DONE ; already thru
                    016E                                                          ; here during $CREATE.
```

```
      ED 04 A8    11    E0  016E          BBS     #FAB$V_UFO,FAB$L_FOP(R8),UFO                  ; branch if UFO
      07 22 A9    05    E1  0173          BBC     #IFB$V_BIO,IFB$B_FAC(R9),10$                  ; branch if not BIO
        00A0 C9   03    93  0178          BITB    #<IFB$M_RU!IFB$M_ONLY_RU>,IFB$B_JNLFLG(R9)        ; don't allow RU BIO
                  DB    12  017D          BNEQ    ERRJNS
                            017F
                            017F     ;
                            017F     ; Next, if the process in which we're executing is a RECOVERY process we
                            017F     ; may not want to journal.  Specifically, if the file we're starting to
                            017F     ; access is one RMS Recovery is recovering, we don't want to
                            017F     ;
                            017F     ;        a. recovery unit journal
                            017F     ;        b. AI or BI journal if we're doing AI recovery
                            017F     ;
                            017F     ; Note: BI recovery must be journaled.  If BI recovery is not journaled,
                            017F     ; the file can be in states never represented by any state representable
                            017F     ; by the RMS journal entries in the journal.  This can happen when a file
                            017F     ; is BI journaled, modified, rolled-back, modified again, and later rolled
                            017F     ; back to a time when first modified.  This is because 'old' record images
                            017F     ; are put in BI journals.  Therefore, a record may get put in the file that
                            017F     ; never shows up in the journal.  Therefore if its backed out by Recovery,
                            017F     ; and recovery is not journaled - that record will never be seen again.
                            017F     ; This problem does not occur with AI journaling because the journal contains
                            017F     ; 'new' record images.
                            017F     ;
                            017F
51     00000000'9F      D0  017F 10$:     MOVL    @#CTL$GL_PCB,R1                               ; get PCB address for test
     16 24 A1     1A    E1  0186          BBC     #PCB$V_RECOVER,PCB$L_STS(R1),20$              ; skip rest if not
                            018B                                                                ;  in RECOVER
          00A1 C9       95  018B          TSTB    IFB$B_RECVRFLGS(R9)                           ; may be in RECOVER, but
                            018F                                                                ;  not recovering this
                            018F                                                                ;  file
                  10    13  018F          BEQL    20$                                           ; branch if not in recovery
                            0191
        00A0 C9   03    8A  0191          BICB    #<IFB$M_RU!IFB$M_ONLY_RU>,IFB$B_JNLFLG(R9)        ; clear RU journalin
     05 00A1 C9   01    E1  0196          BBC     #IFB$V_AI_RECVR,IFB$B_RECVRFLGS(R9),20$ ; skip next if not AI
        00A0 C9   0C    8A  019C          BICB    #<IFB$M_AI!IFB$M_BI>,IFB$B_JNLFLG(R9)     ; clear AI, BI if AI
                            01A1
           07 69  30    E0  01A1 20$:     BBS     #IFB$V_WRTACC,(R9),50$                        ; branch if writing
        00A0 C9   0F    8A  01A5          BICB    #<IFB$M_AI!IFB$M_BI!IFB$M_RU!IFB$M_ONLY_RU>,IFB$B_JNLFLG(R9)
                            01AA                                                                ; clear AI,BI,RU
                  50    11  01AA          BRB     3000$                                         ; branch to AI test.
                            01AC
                            01AC 50$:                                                           ;
     06 00A0 C9   00    E1  01AC 60$:     BBC     #IFB$V_ONLY_RU,IFB$B_JNLFLG(R9),1000$         ; branch if ONLY_RU
                            01B2          SSB     #IFB$V_RU,IFB$B_JNLFLG(R9)                    ; set RU bit
                            01B8
     13 00A0 C9   02    E1  01B8 1000$:   BBC     #IFB$V_BI,IFB$B_JNLFLG(R9),2000$              ; branch if no BI
      53    08C8 CA     7E  01BE          MOVAQ   FWA$Q_BIJNL(R10),R3                           ; BI descriptor
      54    08E0 CA     9E  01C3          MOVAB   FWA$T_BIACE(R10),R4                           ; BI name
            55    02    D0  01C8          MOVL    #CJF$_BI,R5                                   ; indicate BI
             009B       30  01CB          BSBW    OPEN_JNL                                      ; go open channel
           67 50        E9  01CE          BLBC    R0,5000$                                      ; get out on error
                            01D1
     25 00A0 C9   03    E1  01D1 2000$:   BBC     #IFB$V_AI,IFB$B_JNLFLG(R9),3000$              ; branch if no AI
      52     009A 8F    3C  01D7          MOVZWL  #<MJB$C_BLN+RJR$C_EXTLEN>,R2                  ; size of MJB for extend
       000006AA'EF      16  01DC          JSB     RM$ALLOC_MJB                                  ; get the MJB
           53 50        E9  01E2          BLBC    R0,5000$                                      ; get out on error
```

```
         34 A9    51  D0  01E5               MOVL    R1,IFB$L_EXTJNLBUF(R9)                       ; set up pointer
       53   08D0  CA  7E  01E9               MOVAQ   FWA$Q_AIJNL(R10),R3                          ; AI descriptor
       54   08F4  CA  9E  01EE               MOVAB   FWA$T_AIACE(R10),R4                          ; AI name
              55  03  D0  01F3               MOVL    #CJF$_AI,R5                                  ; indicate AI
                0070  30  01F6               BSBW    OPEN_JNL                                     ; go open channel
               3C 50  E9  01F9               BLBC    R0,5000$                                     ; get out on error
                          01FC
      13 00A0 C9  04  E1  01FC      3000$:   BBC     #IFB$V_AT,IFB$B_JNLFLG(R9),4000$             ; branch if no AT
       53   08D8  CA  7E  0202               MOVAQ   FWA$Q_ATJNL(R10),R3                          ; AT descriptor
       54   0908  CA  9E  0207               MOVAB   FWA$T_ATACE(R10),R4                          ; AT name
              55  04  D0  020C               MOVL    #CJF$_AT,R5                                  ; indicate AT
                0057  30  020F               BSBW    OPEN_JNL                                     ; go open channel
               23 50  E9  0212               BLBC    R0,5000$                                     ; get out on error
                          0215
      4A 00A0 C9  01  E1  0215      4000$:   BBC     #IFB$V_RU,IFB$B_JNLFLG(R9),6000$             ; branch if no RU
              55  01  D0  021B               MOVL    #CJF$_RU,R5                                  ; indicate RU
                0048  30  021E               BSBW    OPEN_JNL                                     ; go open channel
               14 50  E9  0221               BLBC    R0,5000$                                     ; return on success
     51    00000000'9F D0 0224               MOVL    @#CTL$GL_RUF,R1                              ; already in RU?
                  38  13  022B               BEQL    6000$                                        ; branch if not
        36 11 A1  01  E1  022D               BBC     #RUCB$V_ACTIVE,RUCB$B_CTRL(R1),7000$
      30 00A2 C9  02  E3  0232               BBCS    #IFB$V_RUP,IFB$B_JNLFLG2(R9),7000$           ; set RU in prog
                          0238                                                                    ; NOTE: Should never
                          0238                                                                    ; fall through
                          0238
            00A0  C9  94  0238      5000$:   CLRB    IFB$B_JNLFLG(R9)                             ; on error clr flgs
     51    50     0C  10  EF  023C           EXTZV   #STS$V_FAC_NO,#STS$S_FAC_NO,R0,R1            ; get error facility
              51  01  D1  0241               CMPL    #RMS$_FACILITY,R1                            ; is error from RMS?
                  22  13  0244               BEQL    7000$                                        ; don't map if so
              52  50  D0  0246               MOVL    R0,R2                                        ; save CJF status
           00000000'EF 16 0249               JSB     RM$MAPERR                                    ; fill in STV
     52    00000000'8F D1 024F               CMPL    #CJF$_NONAME,R2                              ; was error no jnl name?
                  07  12  0256               BNEQ    5010$                                        ; no, use NOJ error
                          0258               RMSERR  JNF                                          ; yes, use JNF error
                  05  11  025D               BRB     5020$                                        ;  and continue
                          025F      5010$:   RMSERR  NOJ                                          ; use NOJ error
                      05  0264      5020$:   RSB                                                  ; return
                          0265
                          0265      6000$:   RMSSUC                                               ; yes, indicate success
                          0268
                      05  0268      7000$:   RSB
```

```
0269         .SBTTL  OPEN_JNL - Common open journal channel
0269
0269  ;++
0269  ; OPEN_JNL - Common open journal channel
0269  ;
0269  ; This routine opens a channel on the specified journal.  It also alocates
0269  ; an RJB if needed.
0269  ;
0269  ; Calling sequence:
0269  ;
0269  ;       BSBW    OPEN_JNL
0269  ;
0269  ; Input Parameters:
0269  ;
0269  ;       R3      Address of Journal Name Descriptor (FWA$Q_xxJNL) (AI,BI,AT only)
0269  ;       R4      Address of Journal Name ACE (FWA$T_xxACE) (AI,BI,AT only)
0269  ;       R5      Journal Type (CJF$_xx)
0269  ;       R9      IFAB address
0269  ;       R10     FWA address
0269  ;       R11     Impure area address
0269  ;
0269  ; Implicit Inputs:
0269  ;
0269  ;       IFB$L_RJB       RJB address
0269  ;       IFB$B_JNLFLG    File's journaling flags
0269  ;       FWA$Q_DEVICE    Device File resides on.
0269  ;       FWA$Q_xxJNL, FWA$T_xxJNLN
0269  ;                       Journal Names for file
0269  ;       FWA$L_UIC       File Owner
0269  ;       FWA$L_PRO       File Protection
0269  ;
0269  ; Output Parameters:
0269  ;
0269  ;       R1-R5           Destroyed
0269  ;
0269  ; Implicit Outputs:
0269  ;
0269  ;       IFB$L_RJB       Address of allocated RJB
0269  ;       IFB$B_JNLFLG2   Files Journaling flags
0269  ;               IFB$V_JNL       Set to indicate journaling initialized.
0269  ;       RJB$W_FLAGS     A bit is set for each channel opened.
0269  ;       RJB$Q_CHAN      One word is filled in with a channel number.
0269  ;
0269  ; Completion Codes:
0269  ;
0269  ;       Any RMS, particualrly, DME,
0269  ;       Any CJF status value from $ASSJNL.
0269  ;
0269  ;
0269  ; Side Effects:
0269  ;
0269  ;       If journaling not previosly initialized on this file, allocates an RJB
0269  ;       for it.
0269  ;
0269  ;--
0269
0269  OPEN_JNL:
```

```
            0559  30 0269              BSBW    RM$ALLOC_RJB_BDB                   ; get journaling BDB/Buffer
           03 50  E8 026C              BLBS    R0,10$                             ; continue if success
            007D  31 026F              BRW     50$                                ; out on error
      52   00A4 C9  D0 0272   10$:     MOVL    IFB$L_RJB(R9),R2                   ; get RJB address
        01    55  D1 0277              CMPL    R5,#CJF$_RU                        ; Opening RU?
              3A  13 027A              BEQL    20$                                ; yes, branch
              63  D4 027C              CLRL    (R3)                               ; set up descriptor
      63    64 04 83 027E              SUBB3   #ACE$T_RMSJNLNAM,(R4),(R3)         ; get length of journal name
              09  14 0282              BGTR    15$                                ; length is >0
  50    00000000'8F D0 0284            MOVL    #CJF$_NONAME,R0                    ; journal not specified
              58  11 028B              BRB     40$                                ; error exit
    04 A3   04 A4 DE 028D     15$:     MOVAL   ACE$T_RMSJNLNAM(R4),4(R3)          ; fill in address of string
                     0292
                     0292              $ASSJNL_S      -                           ; assign journal chan
                     0292                      CHAN = RJB$Q_CHAN-2(R2)[R5], -
                     0292                      JNLTYP = R5, -
                     0292                      JNLNAM = (R3), -
                     0292                      ACMODE = MODE, -
                     0292                      PROT   = FWA$W_PRO(R10), -
                     0292                      OBJUIC = FWA$L_UIC(R10), -
                     0292                      FACCOD = FACILITY
                     02B4
              24  11 02B4              BRB     30$
                     02B6
                     02B6     20$:     $ASSJNL_S      -                           ; open RU chan
                     02B6                      CHAN = RJB$Q_CHAN(R2), -
                     02B6                      JNLTYP = R5, -
                     02B6                      DEVNAM = FWA$Q_DEVICE(R10), -
                     02B6                      ACMODE = MODE, -
                     02B6                      PROT   = FWA$W_PRO(R10), -
                     02B6                      OBJUIC = FWA$L_UIC(R10), -
                     02B6                      FACCOD = FACILITY
                     02DA
           08 50  E9 02DA     30$:     BLBC    R0,40$                             ; return on error
                     02DD
              55  D7 02DD              DECL    R5                                 ; one less than type
                     02DF              SSB     R5,RJB$W_FLAGS(R2)                 ; turn on bit for chan
                  05 02E4              RSB                                        ; return to caller
                     02E5
                     02E5     ;
                     02E5     ; Error Exit
                     02E5     ;
              01  BB 02E5     40$:     PUSHR   #^M<R0>                            ; save R0
      000005F2'EF 16 02E7              JSB     RM$DEAJNL                          ; deallocate RJB
              01  BA 02ED              POPR    #^M<R0>                            ; restore R0
                     02EF
                  05 02EF     50$:     RSB
```

```
                        02F0          .SBTTL  RM$CONJNL - Connect Journal BDB
                        02F0
                        02F0    ;++
                        02F0    ; RM$CONJNL - Connect Journal BDB
                        02F0    ;
                        02F0    ; This routine, called from $CONNECT, builds the necessary data
                        02F0    ; structures onto the IRAB for journaling record processing
                        02F0    ; operations
                        02F0    ;
                        02F0    ; Calling sequence:
                        02F0    ;
                        02F0    ;       BSBW    RM$CONJNL
                        02F0    ;
                        02F0    ; Input Parameters:
                        02F0    ;
                        02F0    ;       R9      Address of IRAB
                        02F0    ;       R10     Address of IFAB
                        02F0    ;       R11     Address of Impure area
                        02F0    ;
                        02F0    ; implicit Inputs:
                        02F0    ;
                        02F0    ;       None.
                        02F0    ;
                        02F0    ; Output Parameters:
                        02F0    ;
                        02F0    ;       R1 - R3,R5      Destroyed
                        02F0    ;       R4              Address of BDB for journaling I/O.
                        02F0    ;
                        02F0    ; Implicit Outputs:
                        02F0    ;
                        02F0    ;       IRB$L_JNLBDB    Address of BDB for journaling I/O.
                        02F0    ;
                        02F0    ; Completion Codes:
                        02F0    ;       Any valid RMS, particualarly DME.
                        02F0    ;
                        02F0    ; Side Effects:
                        02F0    ;       A buffer and BDB are allocated, the BDB is marked perm.
                        02F0    ;
                        02F0    ;--
                        02F0
                        02F0    RM$CONJNL::
                        02F0
                        02F0    ;
                        02F0    ; Determine whether or not we need to allocate a journal BDB and buffer. We
                        02F0    ; only need one if connecting for record access. For block I/O access, simply
                        02F0    ; exit (the journal BDB and buffer will be allocated on the first $WRITE).
                        02F0    ;
                        02F0
         05     E0      02F0            BBS     #IFB$V_BIO,-            ; if we're open for BIO, exit
   0A 22 AA             02F2                    IFB$B_FAC(R10),10$      ;
         06     E1      02F5            BBC     #IFB$V_BRO,-            ; if not opening BRO, we're ok
   08 22 AA             02F7                    IFB$B_FAC(R10),20$      ;  (must be open for record access)
         0B     E1      02FA            BBC     #RAB$V_BIO,-            ; if connecting for record access,
   03 04 A8             02FC                    RAB$L_ROP(R8),20$       ;  we're ok
       007C     31      02FF    10$:    BRW     80$                     ; exit
                        0302
                        0302    ;
```

```
                          0302  ; If the file is sequential, determine the largest probable record size to be
                          0302  ; journaled.  A record can be no larger than the maximum record length.  If
                          0302  ; the MRS was not given, then look at the the longest record length or the
                          0302  ; multiblock count.  If none of these values were specified, then punt.
                          0302  ;
                          0302
                          0302          ASSUME  IFB$C_SEQ EQ 0
                          0302
                23 AA 95  0302  20$:    TSTB    IFB$B_ORGCASE(R10)                      ; is the file sequential?
                   1D 12  0305          BNEQ    50$                                     ; no, use BKS for buffer len
                          0307
             55 60 AA 3C  0307          MOVZWL  IFB$W_MRS(R10),R5                       ; use the max rec. size
                   1F 12  030B          BNEQ    60$                                     ; use it if present
                          030D
             55 52 AA 3C  030D          MOVZWL  IFB$W_LRL(R10),R5                       ; use the LRL for the buffer
                   19 12  0311          BNEQ    60$                                     ; finish buffer size calulat
                          0313
             55 37 A8 9A  0313          MOVZBL  RAB$B_MBC(R8),R5                        ; use the MBC for buffer len
                   04 13  0317          BEQL    30$                                     ; no, buffer will be 1 page
                   67 19  0319          BLSS    ERRMBC                                  ; MBC must be > 0
                   0B 11  031B          BRB     55$
                          031D
          55  0200 8F 3C  031D  30$:    MOVZWL  #512,R5                                 ; buff. will be 1 page
                   08 11  0322          BRB     60$
                          0324
                          0324  ;
                          0324  ; file is not sequential.  Use the bucket size as the buffer length.
                          0324  ;
                          0324
             55 5E AA 9A  0324  50$:    MOVZBL  IFB$B_BKS(R10),R5                       ; get bucket size
          55  55  09 78   0328  55$:    ASHL    #9,R5,R5                                ; convert to bytes
                          032C
      55  00000048 8F C0  032C  60$:    ADDL2   #RJR$C_RECLEN, R5                       ; give some overhead
      55  000001FF 8F C0  0333          ADDL2   #511,R5                                 ; round up to a
      55  000001FF 8F CA  033A          BICL2   #511,R5                                 ; page boundary
                          0341
         00000000'EF 16   0341          JSB     RM$ALDJNLBUF                            ; get BDB and buffer
                37 50 E9  0347          BLBC    R0,90$                                  ; get out on error
                   3E BB  034A          PUSHR   #^M<R1,R2,R3,R4,R5>                     ; save regs zeroed by MOVC5
             51 18 A4 D0  034C          MOVL    BDB$L_ADDR(R4),R1                       ; get RJR address
   61  38  00  61 00 2C   0350          MOVC5   #0,(R1),#0,#RJR$C_HDRLEN,(R1)           ; zero the RJR overhead
                   3E BA  0356          POPR    #^M<R1,R2,R3,R4,R5>                     ; restore regs zeroed by MOV
             30 A9  54 D0 0358          MOVL    R4,IRB$L_JNLBDB(R9)                     ; save BDB addr
                          035C
                          035C          ASSUME  RJR$C_EXTLEN    GT      RJR$C_BLKLEN
                          035C          ASSUME  RJR$C_EXTLEN    GT      RJR$C_AT_RECLEN
                          035C
       1C 00A0 CA  04 E1  035C          BBC     #IFB$V_AT,IFB$B_JNLFLG(R10),80$         ; skip if not AT
          52  009A 8F 3C  0362          MOVZWL  #<MJB$C_BLN+RJR$C_EXTLEN>,R2            ; length of structure
          02  23 AA   91  0367          CMPB    IFB$B_ORGCASE(R10),#IFB$C_IDX           ; indexed file?
                   07 12  036B          BNEQ    70$                                     ; if NEQ no
      52  00000100 8F C0  036D          ADDL    #256,R2                                 ; add in max key size
                 0333 30  0374  70$:    BSBW    RM$ALLOC_MJB                            ; allocate MJB
                07 50 E9  0377          BLBC    R0,90$                                  ; branch if error
             2C A9  51 D0 037A          MOVL    R1,IRB$L_ATJNLBUF(R9)                   ; init pointer
                          037E  80$:    RMSSUC                                          ; indicate success
                      05  0381  90$:    RSB
      0382  1093
      0382  1094 ERRMBC:
      0382  1095         RMSERR  MBC
  05  0387  1096         RSB
```

```
                   0388           .SBTTL  RMSMAPJNL - Write Mapping Entry
                   0388
                   0388   ;++
                   0388   ; RMSMAPJNL - Write Mapping Entry
                   0388   ; RMSMAPJNL_RU - Write RU Mapping Entry
                   0388   ;
                   0388   ; This routine writes a mapping entry into all currently open
                   0388   ; journals for a particular file
                   0388   ;
                   0388   ; Calling sequence:
                   0388   ;
                   0388   ;       BSBW    RMSMAPJNL
                   0388   ;       BSBW    RMSMAPJNL_RU
                   0388   ;
                   0388   ; Input Parameters:
                   0388   ;
                   0388   ;       R8      FAB address (used by COMMON_FILE_AT to write CTX field into RJR)
                   0388   ;       R9      IFAB address
                   0388   ;       R11     Impure area address
                   0388   ;       AP      r0 status till now (I know its a hack, but..) only used for AT
                   0388   ;
                   0388   ; Implicit Inputs:
                   0388   ;
                   0388   ;       IFB$L_RJB       RJB address
                   0388   ;       IFB$L_FWA_PTR   FWA pointer and current contents of FWA
                   0388   ;       RJB$V_OPEN      Set to indicate an open entry; cleared if set.
                   0388   ;       RJB$W_FLAGS     RMS journal channel flags - these will be used
                   0388   ;                       as variable inputs (saved and restored by caller)
                   0388   ;                       to allow AT write at a different time from AI, BI, RU.
                   0388   ;
                   0388   ; Output Parameters:
                   0388   ;
                   0388   ;       R1 - R5         Destroyed
                   0388   ;
                   0388   ; Implicit Outputs:
                   0388   ;
                   0388   ;       RJB$V_OPEN      Cleared if set
                   0388   ;
                   0388   ; Completion Codes:
                   0388   ;
                   0388   ;       Any RMS, particularly DME,
                   0388   ;       CJF -   CJF error, CJF status in STV
                   0388   ;
                   0388   ; Side Effects:
                   0388   ;       May have switched to EXEC AST level.
                   0388   ;
                   0388   ;--
                   0388
                   0388   ;
                   0388   ; Alternate Entry Point for RU handler
                   0388   ;
                   0388
                   0388   RMSMAPJNL_RU::
     01     DD     0388           PUSHL   #1                                    ; indicate RU MAPJNL
     02     11     038A           BRB     MAPJNL
                   038C
                   038C   ;
```

```
                                038C  ; Entry point for AI, BI, AT
                                038C  ;
                                038C  RMS$MAPJNL::
                     7E    D4   038C          CLRL    -(SP)                                   ; indicate not RU MAPJNL
                                038E
                                038E
              7E     56    7D   038E  MAPJNL: MOVQ    R6,-(SP)                                ; save R6, R7
              7E     5A    D0   0391          MOVL    R10,-(SP)                               ; save R10
                                0394
                                0394  ;
                                0394  ; Get RJR buffer address.
                                0394  ;
                   042E    30   0394          BSBW    RM$ALLOC_RJB_BDB                        ; get a journal BDB
                                0397                                                          ; if this is CLOSE
                 03 50     E8   0397          BLBS    R0,10$                                  ; continue if OK
                   009C    31   039A          BRW     80$                                     ; out on error
           5A  30 A9       D0   039D  10$:    MOVL    IFB$L_JNLBDB(R9),R10                    ; first get BDB address
           56  18 AA       D0   03A1          MOVL    BDB$L_ADDR(R10),R6                      ; get RJR address
                                03A5
                                03A5  ;
                                03A5  ; Fill in file name in entry
                                03A5  ;
            5A  38 A9      D0   03A5          MOVL    IFB$L_FWA_PTR(R9),R10                   ; get FWA address
         53   00C4 C6      DE   03A9          MOVAL   RJR$T_FILENAME(R6),R3                   ; get name buff addr
                                03AE
                                03AE          ASSUME  RJR$S_FILENAME EQ 256
                                03AE
                                03AE  ;
                                03AE  ; Set buffer size to 255 because the GETFILNAM code builds a NAM block, etc...
                                03AE  ; and can only cope with a size that fits in a byte.
                                03AE  ;
         54   00FF 8F      3C   03AE          MOVZWL  #<RJR$S_FILENAME-1>,R4                  ; set size of buffer
          00000000'EF      16   03B3          JSB     RM$GETFILNAM                            ; go get file name
          58 A6     54     90   03B9          MOVB    R4,RJR$B_FNS(R6)                        ; put length in entry
                                03BD
                                03BD  ; Fill in header
                                03BD  ;
            54  30 A9      D0   03BD          MOVL    IFB$L_JNLBDB(R9),R4                     ; retrieve jnl BDB addr
      14 A4   01C4 8F      B0   03C1          MOVW    #RJR$C_FILNAMLEN,BDB$W_NUMB(R4)         ; set entry size
         57   00A4 C9      D0   03C7          MOVL    IFB$L_RJB(R9),R7                        ; get RJB address
         03 A6      01     90   03CC          MOVB    #RJR$C_MAPPING,RJR$B_ENTRY_TYPE(R6)     ; fill in file type
         04 A6   23 A9     90   03D0          MOVB    IFB$B_ORGCASE(R9),RJR$B_ORG(R6)         ; fill in org
                 0C AE     D5   03D5          TSTL    ^X0C(SP)                                ; RU call?
                    52     12   03D8          BNEQ    70$                                     ; branch if so
                                03DA
                                03DA          ASSUME  FAB$C_SEQ@-4    EQ      RJR$C_SEQ
                                03DA          ASSUME  FAB$C_REL@-4    EQ      RJR$C_REL
                                03DA          ASSUME  FAB$C_IDX@-4    EQ      RJR$C_IDX
                                03DA
      06 0A A7      04     E5   03DA          BBCC    #RJB$V_OPEN,RJB$W_FLAGS(R7),20$         ; branch if not $OPEN
         05 A6      11     90   03DF          MOVB    #RJR$_OPEN,RJR$B_OPER(R6)               ; fill in operation
                    04     11   03E3          BRB     30$
                                03E5
         05 A6      02     90   03E5  20$:    MOVB    #RJR$_CLOSE,RJR$B_OPER(R6)              ; fill in operation
                                03E9
                                03E9  ; Write indivdual mapping entries
                                03E9  ;
```

```
                       03E9
   54    30 A9    D0   03E9  30$:    MOVL     IFB$L_JNLBDB(R9),R4                        ; restore BDB addr
      7E    53    7D   03ED          MOVQ     R3,-(SP)                                   ; make type and BDB args
                       03F0          RMSSUC                                              ; success if no jnling
09 0A A7    01    E1   03F3          BBC      #RJB$V_BI,RJB$W_FLAGS(R7),40$              ; branch if no BI
      6E    02    9A   03F8          MOVZBL   #CJF$_BI,(SP)                              ; set BI
          004D    30   03FB          BSBW     RM$WRTJNL                                  ; write the record
         26 50    E9   03FE          BLBC     R0,60$                                     ; get out on error
                       0401
09 0A A7    02    E1   0401  40$:    BBC      #RJB$V_AI,RJB$W_FLAGS(R7),50$              ; branch if no AI
      6E    03    9A   0406          MOVZBL   #CJF$_AI,(SP)                              ; set AI
          003F    30   0409          BSBW     RM$WRTJNL                                  ; write the record
         18 50    E9   040C          BLBC     R0,60$                                     ; get out on error
                       040F
13 0A A7    03    E1   040F  50$:    BBC      #RJB$V_AT,RJB$W_FLAGS(R7),60$              ; branch if no AT
      6E    04    9A   0414          MOVZBL   #CJF$_AT,(SP)                              ; set AT
   2C A9    56    D0   0417          MOVL     R6,IFB$L_ATJNLBUF(R9)                      ; shortcut RJR addr.
          04D9    30   041B          BSBW     COMMON_FILE_AT                             ; fill in fields
            2B    10   041E          BSBB     RM$WRTJNL                                  ; write the record
   52    2C A9    D0   0420          MOVL     IFB$L_ATJNLBUF(R9),R2                      ; get RJR address
                       0424          ASSUME   RJR$L_AT_STV     EQ     RJR$L_AT_STS+4
         24 A2    7C   0424          CLRQ     RJR$L_AT_STS(R2)                           ; init status
                       0427
      5E    08    C0   0427  60$:    ADDL2    #8,SP                                      ; clear arglist
            0D    11   042A          BRB      80$                                        ; exit
                       042C
                       042C  ;+
                       042C  ; RU mapping entry.
                       042C  ;-
                       042C
   05 A6    11    90   042C  70$:    MOVB     #RJR$_OPEN,RJR$B_OPER(R6)                  ; fill in operation
            54    DD   0430          PUSHL    R4                                         ; BDB addr
            01    DD   0432          PUSHL    #CJF$_RU                                   ; Set RU
            0C    10   0434          BSBB     RM$WRTJNL_OBJ                              ; write the record
      5E    08    C0   0436          ADDL2    #8,SP                                      ; delete arglist
                       0439
      5A    8E    D0   0439  80$:    MOVL     (SP)+,R10                                  ; restore FWA addr
      56    8E    7D   043C          MOVQ     (SP)+,R6                                   ; restore R6,R7
            8E    D5   043F          TSTL     (SP)+                                      ; clear off call code
                  05   0441          RSB
```

```
                         0442         .SBTTL  RM$WRTJNL - Write Journal Entry
                         0442         .SBTTL  RM$WRTJNL_OBJ - Write Journal Entry with OBJECT_ID Flag
                         0442
                         0442  ;++
                         0442  ; RM$WRTJNL - Write Journal Entry
                         0442  ; RM$WRTJNL_OBJ - Write Journal Entry with OBJECT_ID Flag
                         0442  ;
                         0442  ; This routine fills in the mapping enry sequence number into the
                         0442  ; journaling buffer and then writes it out for either a fab or rab
                         0442  ; operation.
                         0442  ;
                         0442  ; Calling sequence:
                         0442  ;
                         0442  ;       BSBW    RM$WRTJNL
                         0442  ;       BSBW    RM$WRTJNL_OBJ
                         0442  ;
                         0442  ; Input Parameters:
                         0442  ;
                         0442  ;       4(SP)   Type of journal to be written (CJF$_xx)
                         0442  ;       8(SP)   Address of journaling BDB
                         0442  ;       R4      Address of BDB of Related buffer
                         0442  ;       R9      Address of IFB or IRB (depending on call)
                         0442  ;       R10     Address of IFB if IRAB call
                         0442  ;       R11     Address of impure area
                         0442  ;
                         0442  ; Implicit Inputs:
                         0442  ;
                         0442  ;       IFB$L_RJB       Address of RJB
                         0442  ;       RJB$Q_CHAN      One word is used as channel for QIO
                         0442  ;
                         0442  ; Output Parameters:
                         0442  ;
                         0442  ;       R1              Destroyed
                         0442  ;
                         0442  ; Implicit Outputs:
                         0442  ;
                         0442  ;       BDB$T_JNLSEQ    One longword contains new high water mark
                         0442  ;
                         0442  ; Completion Codes:
                         0442  ;
                         0442  ;       CJF     -       CJF error, CJF status in STV
                         0442  ;
                         0442  ; Side Effects:
                         0442  ;       May have switched to EXEC AST level.
                         0442  ;
                         0442  ;--
                         0442
              00000008   0442  RBDB=8          ; stack offset to related BDB address
              0000001C   0442  JTYP=28         ; stack offset to journal type code
              00000020   0442  JBDB=32         ; stack offset to journal BDB
                         0442
                         0442  ;
                         0442  ; Alternate Entry Point to write entry with OBJECT_ID flag.
                         0442  ;
                         0442  RM$WRTJNL_OBJ::
      00FC 8F    BB      0442         PUSHR   #^M<R2,R3,R4,R5,R6,R7>                  ; save regs
        53 08    D0      0446         MOVL    #WRFLG$M_OBJECT_ID,R3                   ; set P6 flags
```

```
                07   11  0449          BRB     WRTJNL
                         044B
                         044B  RMS$WRTJNL::
         00FC 8F     BB  044B          PUSHR   #^M<R2,R3,R4,R5,R6,R7>                          ; save regs
            53 10    D0  044F          MOVL    #WRFLG$M_LOCK,R3                                ; set P6 flags
       52    1C AE   D0  0452  WRTJNL: MOVL    JTYP(SP),R2                                     ; get typ code
       0A    08 A9   91  0456          CMPB    IRB$B_BID(R9),#IRB$C_BID                        ; IRB operation?
                11   13  045A          BEQL    10$                                             ; branch if yes
                         045C
                         045C  ;
                         045C  ; IFAB operation
                         045C  ;
       54    38 A9   D0  045C          MOVL    IFB$L_FWA_PTR(R9),R4                            ; get FWA address
     56    00A4 C9   D0  0460          MOVL    IFB$L_RJB(R9),R6                                ; get RJB address
  11 00A2 C9    02   E0  0465          BBS     #IFB$V_RUP,IFB$B_JNLFLG2(R9),15$                ; branch if RUP
                1C   11  046B          BRB     20$
                         046D
                         046D  ;
                         046D  ; IRAB operation
                         046D  ;
       54    38 AA   D0  046D  10$:    MOVL    IFB$L_FWA_PTR(R10),R4
     56    00A4 CA   D0  0471          MOVL    IFB$L_RJB(R10),R6
  0D 00A2 CA    02   E1  0476          BBC     #IFB$V_RUP,IFB$B_JNLFLG2(R10),20$               ; branch if no RUP
                         047C
                         047C  ;
                         047C  ; IFB, IRB rejoin here if RU in progress.
                         047C  ;
                         047C  15$:    SSB     #WRFLG$V_RUALSO,R3                              ; set RUALSO in P6 flags
             01    52    D1  0480      CMPL    R2,#CJF$_RU                                     ; see if RU write
                04   12  0483          BNEQ    20$                                             ; branch if not
                         0485          SSB     #WRFLG$V_BI,R3                                  ; set RU/BI in P6 flags
                         0489
                         0489  ;
                         0489  ; IFB, IRB rejoin here in no RU in progress
                         0489  ;
       55    20 AE   D0  0489  20$:    MOVL    JBDB(SP),R5                                     ; get jBDB address
                         048D          SSB     #BDB$V_IOP,BDB$B_FLGS(R5)                       ; indicate IO in prog
       51    18 A5   D0  0492          MOVL    BDB$L_ADDR(R5),R1                               ; get buff address
       02 A1    02   90  0496          MOVB    #RJR$C_MAXVER,RJR$B_VERSION(R1)                 ; set journal rec ver #
                3E   BB  049A          PUSHR   #^M<R1,R2,R3,R4,R5>
 08 A1    0920 C4   1C   28  049C      MOVC3   #FWA$S_JNLID,FWA$T_JNLID(R4),RJR$T_JNLID(R1) ; copy journal id
                3E   BA  04A3          POPR    #^M<R1,R2,R3,R4,R5>
       57    14 A5   3C  04A5          MOVZWL  BDB$W_NUMB(R5),R7                               ; get record length
        00000000'EF  16  04A9          JSB     RM$SETEFN                                       ; get EFN
                01   BA  04AF          POPR    #^M<R0>
                         04B1          $QIO_S -                                                ; issue QIO
                         04B1                  EFN     =       R0, -
                         04B1                  CHAN    =       RJB$Q_CHAN-2(R6)[R2], -
                         04B1                  FUNC    =       #IO$_WRITEVBLK, -
                         04B1                  IOSB    =       BDB$L_IOSB(R5), -
                         04B1                  ASTADR  =       RM$STALLAST, -
                         04B1                  ASTPRM  =       R9, -                           ; IRB/IFB
                         04B1                  P1      =       (R1), -                         ; buffer address
                         04B1                  P2      =       R7, -                           ; size of transfer
                         04B1                  P6      =       R3                              ; journal type
             18 50   E9  04D7          BLBC    R0,30$                                          ; get out on error
                         04DA
             00000000'EF   16  04DA  1367          JSB     RM$STALL                               ; wait for completion
             50    48 A5   D0  04E0  1368          MOVL    BDB$L_IOSB(R5),R0                      ; retrieve status
             52    1C AE   D0  04E4  1369          MOVL    JTYP(SP),R2                            ; get typ code
             54    08 AE   D0  04E8  1370          MOVL    RBDB(SP),R4                            ; get related BDB addr
        34 A442    4C A5   D0  04EC  1371          MOVL    BDB$L_IOSB+4(R5),BDB$T_JNLSEQ-4(R4)[R2] ; retrieve seq #
                               04F2  1372  30$:    CSB     #BDB$V_IOP,BDB$B_FLGS(R5)              ; clear IO in prog
                 00FC 8F   BA  04F7  1373          POPR    #^M<R2,R3,R4,R5,R6,R7>                 ; restore regs
                    0B 50  E8  04FB  1374          BLBS    R0,40$                                 ; get out on success
             00000000'EF   16  04FE  1375          JSB     RM$MAPERR                              ; fill in STV
                               0504  1376          RMSERR  CJF                                    ; force CJF error
                           05  0509  1377  40$:    RSB                                            ; return to caller
```

```
                                 050A         .SBTTL  RMS$FRCJNL - Force All Journal Entries for a buffer
                                 050A   ;++
                                 050A   ; FORCE_JNL - Force Journal Entries
                                 050A   ;
                                 050A   ; This routine performs a force operation to all open journals
                                 050A   ; at the high water mark for a buffer.
                                 050A   ;
                                 050A   ; Calling sequence:
                                 050A   ;
                                 050A   ;       BSBW    RMS$FRCJNL
                                 050A   ;
                                 050A   ; Input Parameters:
                                 050A   ;
                                 050A   ;       R4      Address of BDB of Related buffer or
                                 050A   ;               Zero to flush all Entries.
                                 050A   ;       R9      IFAB or IRAB address
                                 050A   ;       R10     IFAB address if IFAB operation
                                 050A   ;       R11     Address of Impure Area
                                 050A   ;
                                 050A   ; Implicit Inputs:
                                 050A   ;
                                 050A   ;       IFB$L_RJB       Address of RJB
                                 050A   ;
                                 050A   ; Output Parameters:
                                 050A   ;
                                 050A   ;       R1 - R3, R5     Destroyed
                                 050A   ;
                                 050A   ; Implicit Outputs:
                                 050A   ;       None.
                                 050A   ;
                                 050A   ; Completion Codes:
                                 050A   ;
                                 050A   ;       CJF - CJF error, Status from QIO in STV
                                 050A   ;
                                 050A   ; Side Effects:
                                 050A   ;       May have switched to EXEC AST level.
                                 050A   ;--
                                 050A
                                 050A   RMS$FRCJNL::
           7E    01      D0      050A           MOVL    #1,-(SP)                                        ; anticipate success
      0A    08   A9      91      050D           CMPB    IRB$B_BID(R9),#IRB$C_BID                        ; IRB operation?
                 07      13      0511           BEQL    10$                                             ; branch if yes
   55     00A4   C9      D0      0513           MOVL    IFB$L_RJB(R9),R5                                ; get RJB address
                 05      11      0518           BRB     15$
   55     00A4   CA      D0      051A   10$:    MOVL    IFB$L_RJB(R10),R5
                                 051F
 0C 0A A5        01      E1      051F   15$:    BBC     #RJB$V_BI,RJB$W_FLAGS(R5),20$                   ; branch if no BI
      52         02      D0      0524           MOVL    #CJF$_BI,R2                                     ; indicate BI
           005A          30      0527           BSBW    FORCE_JNL                                       ; go do force
         03 50           E8      052A           BLBS    R0,20$                                          ; skip on success
      6E         50      D0      052D           MOVL    R0,(SP)                                         ; save error code
                                 0530
 0C 0A A5        02      E1      0530   20$:    BBC     #RJB$V_AI,RJB$W_FLAGS(R5),30$                   ; branch if no AI
      52         03      D0      0535           MOVL    #CJF$_AI,R2                                     ; indicate AI
           0049          30      0538           BSBW    FORCE_JNL                                       ; go do force
         03 50           E8      053B           BLBS    R0,30$                                          ; skip on success
      6E         50      D0      053E           MOVL    R0,(SP)                                         ; save error code
```

```
                        0541
   0C 0A A5    03   E1  0541  30$:   BBC     #RJB$V_AT,RJB$W_FLAGS(R5),40$         ; branch if no AT
          52   04   D0  0546         MOVL    #CJF$_AT,R2                           ; indicate AT
             0038   30  0549         BSBW    FORCE_JNL                             ; go do force
          03 50     E8  054C         BLBS    R0,40$                                ; skip on success
          6E   50   D0  054F         MOVL    R0,(SP)                               ; save error code
                        0552
   1A 0A A5    00   E1  0552  40$:   BBC     #RJB$V_RU,RJB$W_FLAGS(R5),50$         ; branch if no RU
51   00000000'9F    D0  0557         MOVL    @#CTL$GL_RUF,R1                       ; RU in prog?
               11   13  055E         BEQL    50$                                   ; branch if not
   0C 11 A1    01   E1  0560         BBC     #RUCB$V_ACTIVE,RUCB$B_CTRL(R1),50$
          52   01   D0  0565         MOVL    #CJF$_RU,R2                           ; indicate RU
             0019   30  0568         BSBW    FORCE_JNL                             ; go do force
          03 50     E8  056B         BLBS    R0,50$                                ; skip on success
          6E   50   D0  056E         MOVL    R0,(SP)                               ; save error code
                        0571
          50   8E   D0  0571  50$:   MOVL    (SP)+,R0                              ; get worst status
             01 50  E9  0574         BLBC    R0,60$                                ; get out on success
                    05  0577         RSB
     00000000'EF    16  0578  60$:   JSB     RMS$MAPERR                            ; fill in STV
                        057E         RMSERR  CJF                                   ; force CJF error
                    05  0583         RSB
```

```
                      0584          .SBTTL  FORCE_JNL - Force Journal Entries
                      0584
                      0584  ;++
                      0584  ; FORCE_JNL - Force Journal Entries
                      0584  ;
                      0584  ; This routine performs a force operation to the specified journal
                      0584  ; at the high water mark for a buffer.
                      0584  ;
                      0584  ; Calling sequence:
                      0584  ;
                      0584  ;       BSBW    RMSFRCJNL
                      0584  ;
                      0584  ; Input Parameters:
                      0584  ;
                      0584  ;       R2      Type of journal to be forced (CJF$_xx)
                      0584  ;       R4      Address of BDB of Related buffer or
                      0584  ;               Zero to flush all entries.
                      0584  ;       R5      Adddress of RJB
                      0584  ;       R9      IFAB or IRAB address
                      0584  ;       R10     IFAB address if IFAB operation
                      0584  ;       R11     Address of Impure Area
                      0584  ;
                      0584  ; Implicit Inputs:
                      0584  ;
                      0584  ;       IFB$L_RJB       Address of RJB
                      0584  ;       RJB$Q_CHAN      One word is used as channel for QIO
                      0584  ;       BDB$T_JNLSEQ    One longword contains high water mark for force
                      0584  ;
                      0584  ; Output Parameters:
                      0584  ;
                      0584  ;       R0 - R3         Destroyed
                      0584  ;
                      0584  ; Implicit Outputs:
                      0584  ;       None.
                      0584  ;
                      0584  ; Completion Codes:
                      0584  ;
                      0584  ;       Any QIO status value.
                      0584  ;       Any IOSB status vaule from a journaling QIO.
                      0584  ;
                      0584  ; Side Effects:
                      0584  ;       May have switched to EXEC AST level.
                      0584  ;--
                      0584
                      0584  FORCE_JNL:
     50    01    D0   0584          MOVL    #1,R0                                   ; anticipate success
     53    54    D0   0587          MOVL    R4,R3                                   ; see if buffer present
           07    13   058A          BEQL    10$                                     ; branch if not
 53   34 A442    D0   058C          MOVL    BDB$T_JNLSEQ-4(R4)[R2],R3              ; get high water mark
           39    13   0591          BEQL    20$                                     ; if zero, bdb has not
                      0593                                                          ; been used as part of a
                      0593                                                          ; journaling operation.
   00000000'EF   16   0593  10$:    JSB     RM$SETEFN                               ; get EFN
           01    BA   0599          POPR    #^M<R0>
                      059B          $QIO_S -                                        ; issue QIO
                      059B                  EFN     =       R0, -
                      059B                  CHAN    =       RJB$Q_CHAN-2(R5)[R2], -
```

```
                       059B             FUNC    =       #IO$_FORCE, -
                       059B             IOSB    =       IRB$[_IOS(R9), -
                       059B             ASTADR  =       RM$STALLAST, -
                       059B             ASTPRM  =       R9, -
                       059B             P2      =       R3                      ; high water mark
          0A 50   E9   05BF     BLBC    R0,20$                                  ; get out on error
     00000000'EF   16   05C2     JSB     RM$STALL                                ; wait for completion
      50    0C A9   D0   05C8     MOVL    IRB$L_IOS(R9),R0                        ; retrieve status
                       05CC
                 05    05CC 20$:  RSB                                             ; return to caller
```

```
                           05CD          .SBTTL  RM$DSCJNL - Disconnect IRAB Journal Structures
                           05CD
                           05CD  ;++
                           05CD  ; RM$DSCJNL - Disconnect IRAB Journal Structures
                           05CD  ;
                           05CD  ; This routine deallocates the data structures for journaling record
                           05CD  ; processing operations from the IRAB.
                           05CD  ;
                           05CD  ; Calling sequence:
                           05CD  ;
                           05CD  ;       BSBW    RM$DSCJNL
                           05CD  ;
                           05CD  ; Input Parameters:
                           05CD  ;
                           05CD  ;       R9      Address of IRAB
                           05CD  ;       R11     Address of Impure area
                           05CD  ;
                           05CD  ; Implicit Inputs:
                           05CD  ;
                           05CD  ;       IRB$L_JNLBDB    Address of journaling BDB
                           05CD  ;
                           05CD  ; Output Parameters:
                           05CD  ;       R0 - R5         Destroyed
                           05CD  ;
                           05CD  ; Implicit Outputs:
                           05CD  ;       None.
                           05CD  ;
                           05CD  ; Completion Codes:
                           05CD  ;       None.
                           05CD  ;
                           05CD  ; Side Effects:
                           05CD  ;       None.
                           05CD  ;
                           05CD  ;--
                           05CD
                           05CD  RM$DSCJNL::
                           05CD
      54    30 A9    D0    05CD          MOVL    IRB$L_JNLBDB(R9),R4             ; get journal BDB address
               09    13    05D1          BEQL    10$                             ; skip if none
      00000000'EF    16    05D3          JSB     RM$RETJNLBDB                    ; deallocate it
            30 A9    D4    05D9          CLRL    IRB$L_JNLBDB(R9)                ; clear pointer
                           05DC  10$:                                            ;
      54    2C A9    D0    05DC          MOVL    IRB$L_ATJNLBUF(R9),R4           ; get AT MJB address
               0F    13    05E0          BEQL    20$                             ; branch if none
         55    54    D0    05E2          MOVL    R4,R5                           ; copy MJB address for FORCE call
             0188    30    05E5          BSBW    RM$FORCE_MJB                    ; force the IRB AT journaling record
                           05E8                                                  ; Note, errors eaten!
      00000000'EF    16    05E8          JSB     RM$RETBLK1                      ; give it up
            2C A9    D4    05EE          CLRL    IRB$L_ATJNLBUF(R9)              ; clear pointer
                     05    05F1  20$:    RSB
```

```
                              05F2           .SBTTL  RMS$DEAJNL - Close journaling on file
                              05F2
                              05F2    ;++
                              05F2    ; RMS$DEAJNL - Close journaling on file
                              05F2    ;
                              05F2    ; This routine deassigns the journal channels open for the file and
                              05F2    ; deallocates the journaling data structures from the IFAB.
                              05F2    ;
                              05F2    ; Calling sequence:
                              05F2    ;
                              05F2    ;        BSBW    RMS$DEAJNL
                              05F2    ;
                              05F2    ; Input Parameters:
                              05F2    ;
                              05F2    ;        R9      Address of IFAB
                              05F2    ;        R11     Impure area address
                              05F2    ;
                              05F2    ; Implicit Inputs:
                              05F2    ;
                              05F2    ;        IRB$L_RJB       Address of RJB
                              05F2    ;
                              05F2    ; Output Parameters:
                              05F2    ;
                              05F2    ;        R1 - R5         Destroyed
                              05F2    ;
                              05F2    ; Implicit Outputs:
                              05F2    ;        None.
                              05F2    ;
                              05F2    ; Completion Codes:
                              05F2    ;        CJF     - CJF Operation Error, CJF status from $DEASJNL in STV
                              05F2    ;
                              05F2    ; Side Effects:
                              05F2    ;        None.
                              05F2    ;
                              05F2    ;--
                              05F2
                              05F2    RMS$DEAJNL::
                              05F2
          7E    01      D0    05F2           MOVL    #1,-(SP)                        ; assume success
     54    30   A9      D0    05F5           MOVL    IFB$L_JNLBDB(R9),R4             ; jnl BDB/Buffer address
                11      13    05F9           BEQL    2$                              ; skip if none
                5A      DD    05FB           PUSHL   R10                             ; save R10
          5A    59      D0    05FD           MOVL    R9,R10                          ; R10 must be IFAB
        00000000'EF     16    0600           JSB     RMS$RETJNLBDB                   ; deallocate BDB/Buffer
          5A    8E      D0    0606           MOVL    (SP)+,R10                       ; restore R10
          30    A9      D4    0609           CLRL    IFB$L_JNLBDB(R9)                ; clear pointer
          2C    A9      D4    060C    2$:    CLRL    IFB$L_ATJNLBUF(R9)              ; clear shortcut pointer
                              060F                                                   ; to AT RJR
     54    34   A9      D0    060F           MOVL    IFB$L_EXTJNLBUF(R9),R4          ; get extend MJB address
                09      13    0613           BEQL    5$                              ; branch if none
        00000000'EF     16    0615           JSB     RMS$RETBLK1                     ; give it up
          34    A9      D4    061B           CLRL    IFB$L_EXTJNLBUF(R9)             ; clear pointer
     54  00A4   C9      D0    061E    5$:    MOVL    IFB$L_RJB(R9),R4                ; get RJB address
                03      12    0623           BNEQ    7$                              ; skip if none
              006F      31    0625           BRW     45$                             ; get out
   13 0A A4     01      E5    0628    7$:    BBCC    #RJB$V_BI,RJB$W_FLAGS(R4),10$   ; branch if no BI
                              062D           $DEASJNL_S -
```

```
                    062D                 CHAN = RJB$W_BICHAN(R4)
                    063A
        03 50   E8  063A          BLBS   R0,10$                                ; continue on success
       6E   50  D0  063D          MOVL   R0,(SP)                               ; save error code
                    0640
  13 0A A4   02  E5  0640  10$:   BBCC   #RJB$V_AI,RJB$W_FLAGS(R4),20$         ; branch if no AI
                    0645          $DEASJNL_S -                                 ; deassign channel
                    0645                 CHAN = RJB$W_AICHAN(R4)
        03 50   E8  0652          BLBS   R0,20$                                ; continue on success
       6E   50  D0  0655          MOVL   R0,(SP)                               ; save error code
                    0658
  13 0A A4   03  E5  0658  20$:   BBCC   #RJB$V_AT,RJB$W_FLAGS(R4),30$         ; branch if no AT
                    065D          $DEASJNL_S -                                 ; deassign channel
                    065D                 CHAN = RJB$W_ATCHAN(R4)
        03 50   E8  066A          BLBS   R0,30$                                ; continue on success
       6E   50  D0  066D          MOVL   R0,(SP)                               ; save error code
                    0670
  12 0A A4   00  E5  0670  30$:   BBCC   #RJB$V_RU,RJB$W_FLAGS(R4),40$         ; branch if no RU
                    0675          $DEASJNL_S -                                 ; deassign channel
                    0675                 CHAN = RJB$W_RUCHAN(R4)
        03 50   E8  0681          BLBS   R0,40$                                ; continue on success
       6E   50  D0  0684          MOVL   R0,(SP)                               ; save error code
                    0687
        0A A4   B4  0687  40$:    CLRW   RJB$W_FLAGS(R4)                       ; clear open flags
       53   59  D0  068A          MOVL   R9,R3                                 ; deallocate RJB
   00000000'EF  16  068D          JSB    RM$RETBLK                             ;
      00A4 C9   D4  0693          CLRL   IFB$L_RJB(R9)                         ; evaporate pointer
      50    8E  D0  0697  45$:    MOVL   (SP)+,R0                              ; get true error code
        01 50   E9  069A          BLBC   R0,50$                                ; get out on error
            05      069D          RSB
                    069E
   00000000'EF  16  069E  50$:    JSB    RM$MAPERR                             ; set STV
                    06A4          RMSERR CJF                                   ; force CJF error
            05      06A9          RSB                                          ; return to caller
```

```
                                  06AA              .SBTTL  RM$ALLOC_MJB - Alloc and init MJB
                                  06AA
                                  06AA      ;++
                                  06AA      ;
                                  06AA      ; RM$ALLOC_MJB - allocate and initalize a miscellaneous journaling buffer
                                  06AA      ;
                                  06AA      ;       The MJB is used for audit trail entries and AI extend descriptions.
                                  06AA      ;
                                  06AA      ; Calling Sequence:
                                  06AA      ;
                                  06AA      ;       BSBW    RM$ALLOC_MJB
                                  06AA      ;
                                  06AA      ; Input Parameters:
                                  06AA      ;
                                  06AA      ;       R10     IFAB address
                                  06AA      ;       R2      mjb size in bytes
                                  06AA      ;
                                  06AA      ; Output Parameters:
                                  06AA      ;
                                  06AA      ;       R0      status
                                  06AA      ;       R1      MJB address
                                  06AA      ;
                                  06AA      ; Side Effects, Implicit Inputs, Implicit Outputs:
                                  06AA      ;
                                  06AA      ;       None.
                                  06AA      ;
                                  06AA      ;--
                                  06AA
                                  06AA      RM$ALLOC_MJB::
                                  06AA
                                  06AA              ASSUME  <IRB$C_BID&1>   EQ      0
                                  06AA              ASSUME  <IFB$C_BID&1>   EQ      1
                                  06AA              ASSUME  IFB$B_BID       EQ      IRB$B_BID
                                  06AA
                  51    59    D0  06AA              MOVL    R9,R1                                   ; assume ifab addr in r1
               03 08    A9    E8  06AD              BLBS    IFB$B_BID(R9),5$                        ; branch if structure is ifab
                  51    69    D0  06B1              MOVL    IRB$L_IFAB_LNK(R9),R1                   ; get ifab address from irab
                                  06B4      5$:
                  52    07    C0  06B4              ADDL2   #7,R2                                   ; round request up
                  52    07    CA  06B7              BICL2   #7,R2                                   ; ...
          52    52    FE    8F    78  06BA              ASHL    #-2,R2,R2                               ; change bytes to longwords
              00000000'EF    16  06BF              JSB     RM$GETBLK                               ; alloc an MJB on IFB page
                  1A    50    E9  06C5              BLBC    R0,10$                                  ; get out on error
            08 A1    18    90  06C8              MOVB    #MJB$C_BID,MJB$B_BID(R1)                ; identify MJB as MJB
         14 A1    20 A1    DE  06CC              MOVAL   MJB$T_RJR(R1),MJB$L_POINTER(R1)         ; init descriptor
                        3E    BB  06D1              PUSHR   #^M<R1,R2,R3,R4,R5>                     ; save MOVC5 regs
               51    20 A1    DE  06D3              MOVAL   MJB$T_RJR(R1),R1                        ; get RJR address
   61    38    00    61    00    2C  06D7              MOVC5   #0,(R1),#0,#RJR$C_HDRLEN,(R1)           ; zero the RJR overhead
                        3E    BA  06DD              POPR    #^M<R1,R2,R3,R4,R5>                     ; restore MOVC5 regs
                                  06DF              RMSSUC
                        05  06E2      10$:    RSB                                             ; return to caller
                        06E3          .SBTTL  RMS$WRITE_MJB - Write Miscellaneous Journaling Buffer
                        06E3
                        06E3    ;++
                        06E3    ;
                        06E3    ; RMS$WRITE_MJB
                        06E3    ;
                        06E3    ; This routine is used to write a journaling record described by a
                        06E3    ; miscellaneous journaling buffer.
                        06E3    ;
                        06E3    ; Calling Sequence:
                        06E3    ;
                        06E3    ;       BSBW    RMS$WRITE_MJB
                        06E3    ;
                        06E3    ; Input Parameters:
                        06E3    ;
                        06E3    ;       R9      - IFAB or IRAB address
                        06E3    ;       R5      - address of MJB
                        06E3    ;
                        06E3    ; Implicit Inputs:
                        06E3    ;
                        06E3    ;       MJB fields:
                        06E3    ;
                        06E3    ;               JNL     - CJF$_AI, BI, AT, or RU for journal channel to use
                        06E3    ;               FLAGS   - various
                        06E3    ;               DESC    - descriptor of embedded RJR to write
                        06E3    ;
                        06E3    ; Output Parameters:
                        06E3    ;
                        06E3    ;       R0      - status
                        06E3    ;       R6      - destroyed
                        06E3    ;
                        06E3    ; Implicit Outputs:
                        06E3    ;
                        06E3    ;       MJB IOSB has status of operation.
                        06E3    ;
                        06E3    ; Side Effects:
                        06E3    ;
                        06E3    ;       None.
                        06E3    ;
                        06E3    ;--
                        06E3
                        06E3   RMS$WRITE_MJB::
                        06E3
             1C    BB   06E3          PUSHR   #^M<R2,R3,R4>                  ; save work registers
       54    59    D0   06E5          MOVL    R9,R4                          ; get potential IFAB address
                        06E8
                        06E8          ASSUME  IFB$B_BID       EQ      IRB$B_BID
                        06E8
   08    08 A4    91    06E8          CMPB    IFB$B_BID(R4),#IFB$C_BID       ; file or record operation?
             03    13   06EC          BEQL    5$                             ; branch if IFAB
       54    69    D0   06EE          MOVL    IRB$L_IFAB_LNK(R9),R4          ; get IFAB address
                        06F1
  56    00A4 C4    D0   06F1   5$:    MOVL    IFB$L_RJB(R4),R6               ; get pointer to RJB
             5D    13   06F6          BEQL    35$                            ; branch if none
                        06F8
58 0A A5    00    E1    06F8          BBC     #MJB$V_INIT,MJB$W_FLAGS(R5),35$ ; skip if RJR in MJB is useless
                        06FD
```

```
              53   D4  06FD            CLRL    R3                                  ; initialize MODIFIER flags
     04 0A A5  01  E1  06FF            BBC     #MJB$V_FORCE,MJB$W_FLAGS(R5),10$;  skip if not write-thru to jnl
        53  40 8F  90  0704            MOVB    #WRMOD$M_FORCE,R3                   ; indicate write-thru to jnl
                       0708
        52  0C A5  9A  0708    10$:    MOVZBL  MJB$B_JNL(R5),R2                    ; get JNL type for channel calculati
           54  59  D0  070C            MOVL    R9,R4                               ; initialize astparm to IRAB address
     03 0A A5  02  E1  070F            BBC     #MJB$V_FILE,MJB$W_FLAGS(R5),20$     ; branch if assumption OK
           54  5A  D0  0714            MOVL    R10,R4                              ; otherwise astprm is IFAB address
                       0717
       00000000'EF 16  0717    20$:    JSB     RM$SETEFN                           ; get an EFN to wait on
              01   BA  071D            POPR    #^M<R0>                             ; and stick it in R0
                       071F
                       071F            $WRITEJNL_S -
                       071F                    CHAN    = RJB$Q_CHAN-2(R6)[R2], -          ; channel of journal
                       071F                    WRTBUF  = MJB$Q_DESC(R5), -                ; RJR descriptor
                       071F                    MODIF   = R3,-                             ; modifier flags
                       071F                    EFN     = R0,-                             ; event flag to wait on
                       071F                    IOSB    = IRB$L_IOS(R9),-                  ; status of operation
                       071F                    ASTADR  = RM$STALLAST,-                    ; back to RM$STALLAST
                       071F                    ASTPRM  = R4                               ; IFAB or IRAB
                       073F
           21  50  E9  073F            BLBC    R0,50$                              ; go away on error
     14 0A A5  03  E0  0742            BBS     #MJB$V_SYNCH_SHARE,MJB$W_FLAGS(R5),40$   ; branch if SFSB lock
                       0747                                                        ; can't be given up
       00000000'EF 16  0747            JSB     RM$STALL                            ; wait for completion
     18 A5  0C A9  7D  074D    30$:    MOVQ    IRB$L_IOS(R9),MJB$Q_IOSB(R5)        ; save status and seq no in MJB
           0E  50  E9  0752            BLBC    R0,50$                              ; go away on error
                       0755    35$:
              1C   BA  0755            POPR    #^M<R2,R3,R4>                       ; restore registers
                       0757            RMSSUC                                      ; indicate success
              05       075A            RSB                                         ; return to caller
                       075B
       00000000'EF 16  075B    40$:    JSB     RM$STALL_LOCK                       ; wait, keeping file lock (used for
                       0761                                                        ; extend)
              EA   11  0761            BRB     30$                                 ; go check status
                       0763
                       0763    50$:
              1C   BA  0763            POPR    #^M<R2,R3,R4>                       ; restore work registers
                       0765            RMSERR  CJF,R1                              ; default error status
       00000000'EF 17  076A            JMP     RM$MAPERR                           ; map error code and return
                       0770                                                        ; to caller
```

```
                        0770            .SUBTITLE RMS$FORCE_MJB - Force MJB Entries
                        0770   ;++
                        0770   ; RMS$FORCE_MJB
                        0770   ;
                        0770   ;        This routine is called at disconnect to force the journal entries
                        0770   ;        described by the high water mark in the MJB.  (Currently only used
                        0770   ;        for AT record operations.
                        0770   ;
                        0770   ; Inputs:
                        0770   ;        r5      MJB address
                        0770   ;
                        0770   ; Implicit Inputs:
                        0770   ;        contents of the MJB, including MJB$B_JNL and the sequence number
                        0770   ;        in the IOSB.
                        0770   ;
                        0770   ;        rjb has the channel assigned to the AT journal
                        0770   ;
                        0770   ; Outputs:
                        0770   ;        r0 - success or failure
                        0770   ;
                        0770   ; Side Effects:
                        0770   ;
                        0770   ;        AT record journal entries flushed.
                        0770   ;
                        0770   ;--
                        0770
                        0770   RMS$FORCE_MJB::
                        0770
                        0770            RMSSUC                                          ; default to success
              3C   BB   0773            PUSHR   #^M<R2,R3,R4,R5>                        ; save work registers
  54    00A4  CA   D0   0775            MOVL    IFB$L_RJB(R10),R4                       ; get RJB address
              39   13   077A            BEQL    40$                                     ; get out if none
    52    0C  A5   9A   077C            MOVZBL  MJB$B_JNL(R5),R2                        ; get JNL identifier
     00000000'EF   16   0780            JSB     RMS$SETEFN                              ; allocate an event flag
              01   BA   0786            POPR    #^M<R0>                                 ; get EF in R0
                        0788
                        0788            $FORCEJNL_S -
                        0788                    CHAN    = RJB$Q_CHAN-2(R4)[R2], -               ; channel of journal
                        0788                    SEQNO   = MJB$Q_IOSB+4(R5), -                   ; sequence number
                        0788                    EFN     = R0, -                                 ; event flag
                        0788                    IOSB    = IRB$L_IOS(R9), -                      ; use IOSB in IRB
                        0788                    ASTADR  = RMS$STALLAST, -                       ; usual AST address
                        0788                    ASTPRM = R9                                     ; IRAB operation
                        07A4
           11 50   E9   07A4            BLBC    R0,50$                                  ; out on error
     00000000'EF   16   07A7            JSB     RMS$STALL                               ; wait for completion
  18 A5   0C  A9   7D   07AD            MOVQ    IRB$L_IOS(R9),MJB$Q_IOSB(R5)            ; grab status for fun
           03 50   E9   07B2            BLBC    R0,50$                                  ; out on error
              3C   BA   07B5   40$:     POPR    #^M<R2,R3,R4,R5>                        ; restore work registers
              05        07B7            RSB                                             ; return to caller
                        07B8
                        07B8   50$:     RMSERR  CJF,R1                                  ; cjf error
     00000000'EF   16   07BD            JSB     RMS$MAPERR                              ; map the error code
              F0   11   07C3            BRB     40$                                     ; return to caller
```

```
                                    07C5          .SUBTITLE RM$ALLOC_RJB_BDB - Allocate RJB, Journal BDB
                                    07C5  ;++
                                    07C5  ; RM$ALLOC_RJB_BDB
                                    07C5  ;
                                    07C5  ;       This routine allocates an RJB and JNL BDB for use by RMS journaling.
                                    07C5  ;
                                    07C5  ; Inputs:
                                    07C5  ;       R9      IFAB
                                    07C5  ;
                                    07C5  ; Outputs:
                                    07C5  ;       R0      status
                                    07C5  ;       IFB$L_JNLBDB    address of JNL BDB
                                    07C5  ;       IFB$L_RJB       address of RJB
                                    07C5  ;
                                    07C5  ; Side Effects:
                                    07C5  ;       None.
                                    07C5  ;
                                    07C5  ;--
                                    07C5
                                    07C5  RM$ALLOC_RJB_BDB::
                                    07C5
                        38    BB    07C5          PUSHR   #^M<R3,R4,R5>                     ; save work registers
                   00A4 C9    D5    07C7          TSTL    IFB$L_RJB(R9)                     ; RJB present?
                        1E    12    07CB          BNEQ    10$                               ; branch if yes
                   51   59    D0    07CD          MOVL    R9,R1                             ; allocate RJB
                   52   03    D0    07D0          MOVL    #RJB$C_BLN/4,R2                   ; size of RJB
              00000000'EF    16    07D3          JSB     RM$GETBLK                         ; get it
                   61   50    E9    07D9          BLBC    R0,30$                            ; get out on error
              00A4 C9   51    D0    07DC          MOVL    R1,IFB$L_RJB(R9)                  ; save RJB address
                08 A1   16    90    07E1          MOVB    #RJB$C_BID,RJB$B_BID(R1)          ; initialize RJB
                                    07E5          SSB     #IFB$V_JNL,IFB$B_JNLFLG2(R9)      ; indicate RJB present
                30 A9         D5    07EB  10$:    TSTL    IFB$L_JNLBDB(R9)                  ; JNLBDB already allocated?
                        4A    12    07EE          BNEQ    20$                               ; branch if so
                                    07F0
                                    07F0  ;
                                    07F0  ; If AI journaling a relative file - allocate a bigger buffer, on large enough
                                    07F0  ; to contain prolog (512 bytes).
                                    07F0  ;
           0D 00A0 C9   03    E1    07F0          BBC     #IFB$V_AI,IFB$B_JNLFLG(R9),15$  ; skip if not AI journaling
                23 A9   01    91    07F6          CMPB    #IFB$C_REL,IFB$B_ORGCASE(R9)    ; is it relative file?
                        07    12    07FA          BNEQ    15$                             ; branch if not relative
                                    07FC
                                    07FC          ASSUME  <RJR$C_BLKLEN+512> GT RJR$C_FILNAMLEN
                                    07FC
           55    0244 8F    3C    07FC          MOVZWL  #<RJR$C_BLKLEN+512>,R5            ; size of buffer
                        05    11    0801          BRB     16$                               ; join common code
                                    0803  15$:                                              ;
           55    01C4 8F    3C    0803          MOVZWL  #RJR$C_FILNAMLEN,R5               ; size of buffer to allocate
                   7E   5A    D0    0808  16$:    MOVL    R10,-(SP)                         ; save R10, ALDJNLBUF needs R10=IFB
                   5A   59    D0    080B          MOVL    R9,R10                            ; copy IFB address
        55    000001FF 8F    C0    080E          ADDL2   #511,R5                           ; round to page boundary
        55    000001FF 8F    CA    0815          BICL2   #511,R5                           ; ...
              00000000'EF    16    081C          JSB     RM$ALDJNLBUF                      ; allocate jnl BDB and buffer
                   5A   8E    D0    0822          MOVL    (SP)+,R10                         ; restore R10
                   18   50    E9    0825          BLBC    R0,40$                            ; get out on error
                30 A9   54    D0    0828          MOVL    R4,IFB$L_JNLBDB(R9)               ; save address of JNLBDB
                        3E    BB    082C          PUSHR   #^M<R1,R2,R3,R4,R5>               ; save regs zeroed by MOVC5
```

```
                    51    18 A4   D0 082E 1933          MOVL    BDB$L_ADDR(R4),R1               ; get RJR address
     61    38    00    61    00   2C 0832 1934          MOVC5   #0,(R1),#0,#RJR$C_HDRLEN,(R1)   ; zero the RJR overhead
                             3E   BA 0838 1935          POPR    #^M<R1,R2,R3,R4,R5>             ; restore regs zeroed by MOVC5
                                     083A 1936
                                     083A 1937 20$:     RMSSUC                                  ; success
                             38   BA 083D 1938 30$:     POPR    #^M<R3,R4,R5>                   ; restore registers
                                  05 083F 1939          RSB                                     ; to caller
                                     0840 1940 40$:                                             ; deallocate the RJB
                      7E     50   D0 0840 1941          MOVL    R0,-(SP)                        ; save error code
                      53     59   D0 0843 1942          MOVL    R9,R3                           ; address of block holding space
                54    00A4   C9   D0 0846 1943          MOVL    IFB$L_RJB(R9),R4                ; address of RJB
                   00000000'EF    16 084B 1944          JSB     RM$RETBLK                       ; return space and to caller
                      50     8E   D0 0851 1945          MOVL    (SP)+,R0                        ; restore error code
                             38   BA 0854 1946          POPR    #^M<R3,R4,R5>                   ; restore registers
                                  05 0856 1947          RSB                                     ; to caller
```

```
                     0857          .SUBTITLE RM$AT_JNL_RECORD - Write AT Entry for Records
                     0857
                     0857   ;++
                     0857   ; RM$AT_JNL_RECORD
                     0857   ;
                     0857   ;       This routine is responsible for writing any AT journaling record
                     0857   ;       required to describe a record operation.  This routine's primary
                     0857   ;       task is to make sure the RJR overhead is filled in properly, and
                     0857   ;       the correct MJB inputs are set.  RM$WRITE_MJB is then called to
                     0857   ;       actually perform the CJF write service.
                     0857   ;
                     0857   ; Calling Sequence:
                     0857   ;
                     0857   ;       BSBW    RM$AT_JNL_RECORD
                     0857   ;
                     0857   ;       This routine is called only by RM$EXRMS.
                     0857   ;
                     0857   ; Input Parameters:
                     0857   ;
                     0857   ;       R0      operation status to this point
                     0857   ;       R8      user's RAB
                     0857   ;       R9      IRAB
                     0857   ;       R10     IFAB
                     0857   ;
                     0857   ; Implicit Inputs:
                     0857   ;
                     0857   ;       IRB$L_ATJNLBUF  - pointer to MJB containing RJR
                     0857   ;       RJR$B_OPER      - must be filled in by caller
                     0857   ;
                     0857   ; Output Parameters:
                     0857   ;
                     0857   ;       r0      operation status
                     0857   ;       r1      destroyed
                     0857   ;
                     0857   ; Implicit Outputs:
                     0857   ;
                     0857   ;       None.   (for now)
                     0857   ;
                     0857   ; Side Effects:
                     0857   ;
                     0857   ;       RJR written to CJF
                     0857   ;
                     0857   ;--
                     0857
                     0857  RM$AT_JNL_RECORD::
                     0857
              59  D5 0857          TSTL    R9                      ; any structure address?
              01  12 0859          BNEQ    2$                      ; if no, must be structureless exit
                  05 085B  1$:     RSB                             ; nothing to do
                     085C
                     085C          ASSUME  IFB$B_BID       EQ      IRB$B_BID
                     085C
      0A    08 A9  91 085C  2$:     CMPB    IFB$B_BID(R9),#IRB$C_BID ; is this an IRAB?
              F9  12 0860          BNEQ    1$                      ; if neq no, forget it
                     0862
 F3 00A0 CA    04  E1 0862          BBC     #IFB$V_AT,IFB$B_JNLFLG(R10),1$ ; skip if not AT journaling
              30  BB 0868          PUSHR   #^M<R4,R5>              ; save work registers
```

```
             55    2C A9   D0 086A          MOVL    IRB$L_ATJNLBUF(R9),R5     ; get MJB address
                      67   13 086E          BEQL    70$                       ; skip if none
                              0870
                              0870    ;
                              0870    ; Fill in required MJB fields
                              0870    ;
          0C A5       04   90 0870          MOVB    #CJF$_AT,MJB$B_JNL(R5)    ; indicate we're audit trail journaling
                   0A A5   B4 0874          CLRW    MJB$W_FLAGS(R5)           ; nothing special for WRITEJNL call
       10 A5     004C 8F   3C 0877          MOVZWL  #RJR$C_AT_RECLEN,MJB$Q_DESC(R5) ; set up record length
                              087D
             54    20 A5   DE 087D          MOVAL   MJB$T_RJR(R5),R4          ; get RJR address in R4
                   05 A4   D5 0881          TSTL    RJR$B_OPER(R4)                    ; any op specified?
                      51   13 0884          BEQL    70$                               ; skip if none
        4F 0A A5      00   E3 0886          BBCS    #MJB$V_INIT,MJB$W_FLAGS(R5),90$   ; skip filling in RJR if already
                              088B                                                    ; done
                              088B    10$:                                            ; RJR overhead filled in
          24 A4       50   D0 088B          MOVL    R0,RJR$L_AT_STS(R4)               ; get status
                              088F          SSB     #16,RJR$L_AT_STS(R4)              ; make it an RMS status
       28 A4       0C A8   D0 0894          MOVL    RAB$L_STV(R8),RJR$L_AT_STV(R4)    ; and get STV also
                              0899    ;
                              0899    ; Pull user's request from RAB into journal record.  Must probe structures.
                              0899    ; All relevant data that was available at the beginning of the operation
                              0899    ; is already in the journal record.  It was put there by RM$AT_COM_RAB.
                              0899    ;
                      58   D5 0899    20$:  TSTL    R8                                ; user structure?
                      17   13 089B          BEQL    60$                               ; branch if no RAB
                              089D          IFNORD  #RAB$C_BLN,(R8),60$               ; skip rest if not readable
               01     68   91 08A5          CMPB    (R8),#RAB$C_BID                   ; is it a RAB?
                      0A   12 08A8          BNEQ    60$                               ; branch if no RAB
                              08AA
                              08AA    ;
                              08AA    ; We found a readable RAB, now fill AT entry in with the RAB contents.
                              08AA    ;
       44 A4       10 A8   D0 08AA          MOVL    RAB$L_RFA0(R8),RJR$L_AT_RFA0(R4);  1st part of RFA
       48 A4       14 A8   B0 08AF          MOVW    RAB$W_RFA4(R8),RJR$W_AT_RFA4(R4);  2nd part of RFA
                              08B4
          51    41 A4      9A 08B4    60$:  MOVZBL  RJR$B_AT_KSZ(R4),R1               ; get key size
          10 A5       51   C0 08B8          ADDL2   R1,MJB$Q_DESC(R5)                 ; account for key size
                    FE24   30 08BC          BSBW    RM$WRITE_MJB                      ; write the AT record
                              08BF
                              08BF          ASSUME  RJR$L_AT_STV    EQ      RJR$L_AT_STS+4
                              08BF
                   24 A4   7C 08BF          CLRQ    RJR$L_AT_STS(R4)                  ; init status for next time
                   05 A4   94 08C2          CLRB    RJR$B_OPER(R4)                    ; and operation
                              08C5
                              08C5    ;
                              08C5    ; Now zero search KEY so it doesn't linger in the buffer.
                              08C5    ;
          51    41 A4      9A 08C5          MOVZBL  RJR$B_AT_KSZ(R4),R1               ; get key size for clear
                      0C   13 08C9          BEQL    70$                               ; skip if none
                      0F   BB 08CB          PUSHR   #^M<R0,R1,R2,R3>                  ; save MOVC3 registers
 4C A4   51    00   4C A4  00   2C 08CD     MOVC5   #0,RJR$T_AT_KEY(R4),#0,R1,-       ; zero out KEY for next time
                              08D5                  RJR$T_AT_KEY(R4)                  ;
                      0F   BA 08D5          POPR    #^M<R0,R1,R2,R3>                  ; restore MOVC3 registers
                              08D7
                      30   BA 08D7    70$:  POPR    #^M<R4,R5>                        ; restore work registers
                      05      08D9    80$:  RSB                               ; return to caller
```

```
                                 08DA
                                 08DA   90$:                                      ; fill in RJR overhead
                                 08DA
                                 08DA          ASSUME RJR$B_ENTRY_TYPE EQ <RJR$B_VERSION+1>
                                 08DA
                 0602 8F    B0   08DA          MOVW    #<<RJR$C_AT_RECORD@8>+RJR$C_MAXVER>,-
                   02 A4         08DE                  RJR$B_VERSION(R4)                ; version, type
           04 A4   23 AA    90   08E0          MOVB    IFB$B_ORGCASE(R10),RJR$B_ORG(R4); file organization
                      3F    BB   08E5          PUSHR   #^M<R0,R1,R2,R3,R4,R5>           ; save registers MOVC3 destroys
              55   38 AA    D0   08E7          MOVL    IFB$L_FWA_PTR(R10),R5            ; get FWA address
      08 A4   0920 C5  1C   28   08EB          MOVC3   #FWA$S_JNLID,FWA$T_JNLID(R5),RJR$T_JNLID(R4) ; journal id
                      3F    BA   08F2          POPR    #^M<R0,R1,R2,R3,R4,R5>           ; restore MOVC3 registers
                    FF94    31   08F4          BRW     10$                              ; join common code
```

```
                        08F7        .SUBTITLE COMMON_FILE_AT - Get common AT file data
                        08F7  ;++
                        08F7  ; COMMON_FILE_AT
                        08F7  ;
                        08F7  ;     This routine is used to fill in the AT journal entry with data from the
                        08F7  ;     IFAB at MAPJNL time.
                        08F7  ;
                        08F7  ; Inputs:
                        08F7  ;
                        08F7  ;     r8      FAB
                        08F7  ;     r9      IFAB
                        08F7  ;
                        08F7  ; Outputs:
                        08F7  ;
                        08F7  ;     AT journal record fields filled in.
                        08F7  ;
                        08F7  ; Side Effects:
                        08F7  ;
                        08F7  ;     Currently, the STS/STV is forced to success due to difficulties
                        08F7  ;     in acquiring the info when the journal entry must be written.
                        08F7  ;     (IE,, can't do it at exit RMS like record operations because
                        08F7  ;     data structures must be deallocated at release time. Better
                        08F7  ;     solution is to make file AT info hendled by an MJB also, and write
                        08F7  ;     and deallocate the file MJB at exit RMS.)
                        08F7  ;--
                        08F7
                        08F7  COMMON_FILE_AT:
                        08F7
             04     BB  08F7        PUSHR   #^M<R2>                 ; save work register
     52   2C A9     D0  08F9        MOVL    IFB$L_ATJNLBUF(R9),R2   ; get address of journal record (RJR)
                        08FD
  5A A2   22 A9     90  08FD        MOVB    IFB$B_FAC(R9),RJR$B_FAC(R2) ; fill in specified file access
  5B A2   4E A9     90  0902        MOVB    IFB$B_SHR(R9),RJR$B_SHR(R2) ; fill in specified file sharing
  48 A2   70 A9     D0  0907        MOVL    IFB$L_HBK(R9),RJR$L_ALLOC(R2) ; fill in high allocation
  24 A2   08 A8     D0  090C        MOVL    FAB$L_STS(R8),RJR$L_AT_STS(R2)  ; status
  28 A2   0C A8     D0  0911        MOVL    FAB$L_STV(R8),RJR$L_AT_STV(R2)  ; STV
  2C A2   18 A8     D0  0916        MOVL    FAB$L_CTX(R8),RJR$L_AT_CTX(R2)  ; User definable CTX field
                        091B
             04     BA  091B  10$:  POPR    #^M<R2>                 ; restore work register
             05         091D        RSB                             ; to RMS$MAPJNL
```

```
                        091E                .SUBTITLE RMS$AT_COM_RAB - Get common AT record data
                        091E        ;++
                        091E        ; AT_COM_RAB
                        091E        ;
                        091E        ;       This routine scarfs up and puts in the RMS journaling record the
                        091E        ;       common RAB data at the beginning of an operation.
                        091E        ;
                        091E        ; Inputs:
                        091E        ;
                        091E        ;       R1      rjr operation id
                        091E        ;       R8      RAB (the sucker is assumed to be probed.)
                        091E        ;       R9      irab
                        091E        ;       R10     ifab
                        091E        ;
                        091E        ; Outputs:
                        091E        ;
                        091E        ;       Some AT record RJR fields filled in.
                        091E        ;
                        091E        ;--
                        091E
                        091E        RMS$AT_COM_RAB::
                        091E
              10    BB  091E                PUSHR   #^M<R4>                         ; save work register
      54   2C A9    D0  0920                MOVL    IRB$L_ATJNLBUF(R9),R4           ; get MJB address
              3C    13  0924                BEQL    60$                             ; skip if none
                        0926
      54   20 A4    DE  0926                MOVAL   MJB$T_RJR(R4),R4                ; get RJR address in R4
                        092A
   3C A4   04 A8    D0  092A                MOVL    RAB$L_ROP(R8),RJR$L_AT_ROP(R4)  ; user's ROP
   40 A4   35 A8    90  092F                MOVB    RAB$B_KRF(R8),RJR$B_AT_KRF(R4)  ; user's key of reference
   42 A4   1E A8    90  0934                MOVB    RAB$B_RAC(R8),RJR$B_AT_RAC(R4)  ; user's record access
      05 A4   51    90  0939                MOVB    R1,RJR$B_OPER(R4)               ; operation code
   2C A4   18 A8    D0  093D                MOVL    RAB$L_CTX(R8),RJR$L_AT_CTX(R4)  ; User context field
                        0942
                        0942        ;
                        0942        ; Probe key buffer before getting key.
                        0942        ;
      01   1E A8    91  0942                CMPB    RAB$B_RAC(R8),#RAB$C_KEY        ; keyed access?
              1A    12  0946                BNEQ    60$                             ; if not, no key size
   41 A4   34 A8    90  0948                MOVB    RAB$B_KSZ(R8),RJR$B_AT_KSZ(R4)  ; user's key size
              13    13  094D                BEQL    60$                             ; if zero, no key
                        094F                IFNORD  RJR$B_AT_KSZ(R4),RAB$L_KBF(R8),60$ ; skip if can't get keybuffer
                        0957
                        0957        ;
                        0957        ; Copy search key into journal record
                        0957        ;
              3E    BB  0957                PUSHR   #^M<R1,R2,R3,R4,R5>             ; save MOVC3 registers
           41 A4    28  0959                MOVC3   RJR$B_AT_KSZ(R4),-              ; move KEY_SIZE number of chars
           30 B8        095C                        @RAB$L_KBF(R8),-                ;   from rab keybuffer
           4C A4        095E                        RJR$T_AT_KEY(R4)                ;   to journal record
              3E    BA  0960                POPR    #^M<R1,R2,R3,R4,R5>             ; restore MOVC3 egisters
                        0962
              10    BA  0962        60$:    POPR    #^M<R4>                         ; restore work register
                    05  0964        70$:    RSB                                     ; to caller
                        0965
                        0965                .END
```

```
                                                              I  3
RMOJOURNL                           RMS Journaling Manager             16-SEP-1984 00:25:13  VAX/VMS Macro V04-00          Page  48
Symbol table                                                            5-SEP-1984 16:21:57  [RMS.SRC]RMOJOURNL.MAR;1            (22)

$$.PSECT_EP                      = 00000000                  FWA$Q_DEVICE                  = 000000E0
$$RMSTEST                        = 0000001A                  FWA$Q_ID_DATE                 = 00000934
$$RMS_PBUGCHK                    = 00000010                  FWA$S_AIACE                   = 00000014
$$RMS_TBUGCHK                    = 00000008                  FWA$S_ATACE                   = 00000014
$$RMS_UMODE                      = 00000004                  FWA$S_BIACE                   = 00000014
$$T1                             = 00000000                  FWA$S_BIJNLN                  = 00000010
ACE$B_TYPE                       = 00000001                  FWA$S_IDACE                   = 00000020
ACE$C_AIJNL                      = 00000003                  FWA$S_JNLID                   = 0000001C
ACE$C_ATJNL                      = 00000004                  FWA$T_AIACE                   = 000008F4
ACE$C_BIJNL                      = 00000002                  FWA$T_ATACE                   = 00000908
ACE$C_JNLID                      = 00000008                  FWA$T_BIACE                   = 000008E0
ACE$M_HIDDEN                     = 00000400                  FWA$T_FIBBUF                  = 000001F4
ACE$M_NOPROPAGATE                = 00000800                  FWA$T_FID                     = 0000092C
ACE$M_PROTECTED                  = 00000200                  FWA$T_IDACE                   = 0000091C
ACE$T_RMSJNLNAM                  = 00000004                  FWA$T_JNLID                   = 00000920
ACE$W_FLAGS                      = 00000002                  FWA$W_PRO                     = 0000002C
ASS_DONE                           00000164 R      01        GET_JNL                         000000A1 R      01
ATR$C_ADDACLENT                  = 0000001F                  IFB$B_BID                     = 00000008
ATR$C_FNDACLTYP                  = 00000023                  IFB$B_BKS                     = 0000005E
ATR$C_JOURNAL                    = 0000001D                  IFB$B_FAC                     = 00000022
ATR$C_UIC_RO                     = 0000001A                  IFB$B_JNLFLG                  = 000000A0
BDB$B_FLGS                       = 0000000A                  IFB$B_JNLFLG2                 = 000000A2
BDB$L_ADDR                       = 00000018                  IFB$B_ORGCASE                 = 00000023
BDB$L_IOSB                       = 00000048                  IFB$B_RECVRFLGS               = 000000A1
BDB$T_JNLSEQ                     = 00000038                  IFB$B_SHR                     = 0000004E
BDB$V_IOP                        = 00000002                  IFB$C_BID                     = 0000000B
BDB$W_NUMB                       = 00000014                  IFB$C_IDX                     = 00000002
CJF$ASSJNL                         ********  GX    01        IFB$C_REL                     = 00000001
CJF$DEASJNL                        ********  GX    01        IFB$C_SEQ                     = 00000000
CJF$FORCEJNL                       ********  GX    01        IFB$L_ATJNLBUF                = 0000002C
CJF$GETJNL                         ********  GX    01        IFB$L_EXTJNLBUF               = 00000034
CJF$WRITEJNL                       ********  GX    01        IFB$L_FWA_PTR                 = 00000038
CJF$_AI                          = 00000003                  IFB$L_HBK                     = 00000070
CJF$_AT                          = 00000004                  IFB$L_JNLBDB                  = 00000030
CJF$_BI                          = 00000002                  IFB$L_RJB                     = 000000A4
CJF$_NONAME                        ********   X    01        IFB$M_AI                      = 00000008
CJF$_RU                          = 00000001                  IFB$M_BI                      = 00000004
COMMON_FILE_AT                     000008F7 R      01        IFB$M_ONLY_RU                 = 00000001
CTL$GL_PCB                         ********   X    01        IFB$M_RU                      = 00000002
CTL$GL_RUF                         ********   X    01        IFB$V_AI                      = 00000003
ERRJNS                             0000015A R      01        IFB$V_AI_RECVR                = 00000001
ERRMBC                             00000382 R      01        IFB$V_AT                      = 00000004
FAB$C_IDX                        = 00000020                  IFB$V_BI                      = 00000002
FAB$C_REL                        = 00000010                  IFB$V_BIO                     = 00000005
FAB$C_SEQ                        = 00000000                  IFB$V_BRO                     = 00000006
FAB$L_CTX                        = 00000018                  IFB$V_DONE_ASS_JNL            = 00000004
FAB$L_FOP                        = 00000004                  IFB$V_JNL                     = 00000001
FAB$L_STS                        = 00000008                  IFB$V_ONLY_RU                 = 00000000
FAB$L_STV                        = 0000000C                  IFB$V_RU                      = 00000001
FAB$V_UFO                        = 00000011                  IFB$V_RUP                     = 00000002
FACILITY                           00000000 R      01        IFB$V_WRTACC                  = 00000030
FIB$W_FID                        = 00000004                  IFB$W_LRL                     = 00000052
FORCE_JNL                          00000584 R      01        IFB$W_MRS                     = 00000060
FWA$L_UIC                        = 00000028                  IO$_FORCE                     = 00000037
FWA$Q_AIJNL                      = 000008D0                  IO$_WRITEVBLK                 = 00000030
FWA$Q_ATJNL                      = 000008D8                  IRB$B_BID                     = 00000008
FWA$Q_BIJNL                      = 000008C8                  IRB$C_BID                     = 0000000A
```

```
                                                                  J  3
RMOJOURNL                        RMS Journaling Manager                        16-SEP-1984 00:25:13  VAX/VMS Macro V04-00          Page  49
Symbol table                                                                    5-SEP-1984 16:21:57  [RMS.SRC]RMOJOURNL.MAR;1           (22)

IRB$L_ATJNLBUF                   = 0000002C                     RJR$B_AT_RAC                    = 00000042
IRB$L_IFAB_LNK                   = 00000000                     RJR$B_ENTRY_TYPE                = 00000003
IRB$L_IOS                        = 0000000C                     RJR$B_FAC                       = 0000005A
IRB$L_JNLBDB                     = 00000030                     RJR$B_FNS                       = 00000058
JBDB                             = 00000020                     RJR$B_OPER                      = 00000005
JTYP                             = 0000001C                     RJR$B_ORG                       = 00000004
MAPJNL                             0000038E R       01          RJR$B_SHR                       = 0000005B
MJB$B_BID                        = 00000008                     RJR$B_VERSION                   = 00000002
MJB$B_JNL                        = 0000000C                     RJR$C_AT_RECLEN                 = 0000004C
MJB$C_BID                        = 00000018                     RJR$C_AT_RECORD                 = 00000006
MJB$C_BLN                        = 00000020                     RJR$C_BLKLEN                    = 00000044
MJB$L_POINTER                    = 00000014                     RJR$C_EXTLEN                    = 0000007A
MJB$Q_DESC                       = 00000010                     RJR$C_FILNAMLEN                 = 000001C4
MJB$Q_IOSB                       = 00000018                     RJR$C_HDRLEN                    = 00000038
MJB$T_RJR                        = 00000020                     RJR$C_IDX                       = 00000002
MJB$V_FILE                       = 00000002                     RJR$C_MAPPING                   = 00000001
MJB$V_FORCE                      = 00000001                     RJR$C_MAXVER                    = 00000002
MJB$V_INIT                       = 00000000                     RJR$C_RECLEN                    = 00000048
MJB$V_SYNCH_SHARE                = 00000003                     RJR$C_REL                       = 00000001
MJB$W_FLAGS                      = 0000000A                     RJR$C_SEQ                       = 00000000
MODE                               00000002 R       01          RJR$L_ALLOC                     = 00000048
OPEN_JNL                           00000269 R       01          RJR$L_AT_CTX                    = 0000002C
PCB$L_STS                        = 00000024                     RJR$L_AT_RFA0                   = 00000044
PCB$L_UIC                        = 000000BC                     RJR$L_AT_ROP                    = 0000003C
PCB$V_RECOVER                    = 0000001A                     RJR$L_AT_STS                    = 00000024
PSL$C_EXEC                       = 00000001                     RJR$L_AT_STV                    = 00000028
RAB$B_KRF                        = 00000035                     RJR$S_FILENAME                  = 00000100
RAB$B_KSZ                        = 00000034                     RJR$T_AT_KEY                    = 0000004C
RAB$B_MBC                        = 00000037                     RJR$T_FILENAME                  = 000000C4
RAB$B_RAC                        = 0000001E                     RJR$T_JNLID                     = 00000008
RAB$C_BID                        = 00000001                     RJR$W_AT_RFA4                   = 00000048
RAB$C_BLN                        = 00000044                     RJR$_CLOSE                      = 00000002
RAB$C_KEY                        = 00000001                     RJR$_OPEN                       = 00000011
RAB$L_CTX                        = 00000018                     RMS$ALDJNLBUF                     ********    X   01
RAB$L_KBF                        = 00000030                     RMS$ALLOC_MJB                     000006AA RG     01
RAB$L_RFA0                       = 00000010                     RMS$ALLOC_RJB_BDB                 000007C5 RG     01
RAB$L_ROP                        = 00000004                     RMS$ASSJNL                        00000168 RG     01
RAB$L_STV                        = 0000000C                     RMS$AT_COM_RAB                    0000091E RG     01
RAB$V_BIO                        = 0000000B                     RMS$AT_JNL_RECORD                 00000857 RG     01
RAB$W_RFA4                       = 00000014                     RMS$CONJNL                        000002F0 RG     01
RBDB                             = 00000008                     RMS$DEAJNL                        000005F2 RG     01
RJB$B_BID                        = 00000008                     RMS$DSCJNL                        000005CD RG     01
RJB$C_BID                        = 00000016                     RMS$FORCE_MJB                     00000770 RG     01
RJB$C_BLN                        = 0000000C                     RMS$FRCJNL                        0000050A RG     01
RJB$Q_CHAN                       = 00000000                     RMS$GETBLK                        ********    X   01
RJB$V_AI                         = 00000002                     RMS$GETFILNAM                     ********    X   01
RJB$V_AT                         = 00000003                     RMS$GETJNL                        00000004 RG     01
RJB$V_BI                         = 00000001                     RMS$MAPERR                        ********    X   01
RJB$V_OPEN                       = 00000004                     RMS$MAPJNL                        0000038C RG     01
RJB$V_RU                         = 00000000                     RMS$MAPJNL_RU                     00000388 RG     01
RJB$W_AICHAN                     = 00000004                     RMS$RETBLK                        ********    X   01
RJB$W_ATCHAN                     = 00000006                     RMS$RETBLK1                       ********    X   01
RJB$W_BICHAN                     = 00000002                     RMS$RETJNLBDB                     ********    X   01
RJB$W_FLAGS                      = 0000000A                     RMS$RTVJNL                        000000F5 RG     01
RJB$W_RUCHAN                     = 00000000                     RMS$SETEFN                        ********    X   01
RJR$B_AT_KRF                     = 00000040                     RMS$STALL                         ********    X   01
RJR$B_AT_KSZ                     = 00000041                     RMS$STALLAST                      ********    X   01
```

```
                                                                    K 3
RMOJOURNL                         RMS Journaling Manager                          16-SEP-1984 00:25:13  VAX/VMS Macro V04-00          Page  50
Symbol table                                                                       5-SEP-1984 16:21:57  [RMS.SRC]RMOJOURNL.MAR;1            (22)

RM$STALL_LOCK                       ********   X    01
RM$WRITE_MJB                        000006E3 RG     01
RM$WRTJNL                           0000044B RG     01
RM$WRTJNL_OBJ                       00000442 RG     01
RMS$_CJF                          = 0001C164
RMS$_FACILITY                     = 00000001
RMS$_JNF                          = 0001C052
RMS$_JNS                          = 000187F4
RMS$_MBC                          = 00018734
RMS$_NOJ                          = 0001C154
RUCB$B_CTRL                       = 00000011
RUCB$V_ACTIVE                     = 00000001
STS$S_FAC_NO                      = 0000000C
STS$V_FAC_NO                      = 00000010
SYS$GETTIM                          ********  GX    01
SYS$QIO                             ********  GX    01
UFO                                 00000160 R      01
WRFLG$M_LOCK                      = 00000010
WRFLG$M_OBJECT_ID                 = 00000008
WRFLG$V_BI                        = 00000001
WRFLG$V_RUALSO                    = 00000002
WRMOD$M_FORCE                     = 00000040
WRTJNL                              00000452 R      01

                                                  +----------------+
                                                  ! Psect synopsis !
                                                  +----------------+

PSECT name                        Allocation          PSECT No.  Attributes
----------                        ----------          ---------  ----------
.  ABS  .                         00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
RM$RMS_JOURNAL                    00000965  ( 2405.)  01 (  1.)    PIC   USR   CON   REL   GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                             00000000  (    0.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                               +------------------------+
                                               ! Performance indicators !
                                               +------------------------+

Phase                        Page faults    CPU Time        Elapsed Time
-----                        -----------    --------        ------------
Initialization                        30    00:00:00.05     00:00:00.93
Command processing                   119    00:00:00.68     00:00:04.60
Pass 1                               721    00:00:34.00     00:01:34.62
Symbol table sort                      0    00:00:04.93     00:00:08.20
Pass 2                               367    00:00:07.75     00:00:18.28
Symbol table output                   29    00:00:00.26     00:00:00.73
Psect synopsis output                  2    00:00:00.04     00:00:00.11
Cross-reference output                 0    00:00:00.00     00:00:00.00
Assembler run totals                1270    00:00:47.71     00:02:07.48

The working set limit was 2400 pages.
189336 bytes (370 pages) of virtual memory were used to buffer the intermediate code.
There were 170 pages of symbol table space allocated to hold 3235 non-local and 104 local symbols.
2173 source lines were read in Pass 1, producing 20 object records in Pass 2.
51 pages of virtual memory were used to define 50 macros.
```

```
                                   +----------------------------+
                                   ! Macro library statistics !
                                   +----------------------------+

Macro library name                  Macros defined
------------------                  --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                  16
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                   4
_$255$DUA28:[SYSLIB]STARLET.MLB;2               26
TOTALS (all libraries)                          46

3505 GETS were required to define 46 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RMOJOURNL/OBJ=OBJ$:RMOJOURNL MSRC$:RMOJOURNL/UPDATE=(ENH$:RMOJOURNL)+EXECML$/LIB+LIB$:RMS/LIB
```

0318 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RM0EXTRMS
LIS

RM0DIRSCN
LIS

RM0CRECOM
LIS

RM0FABCHK
LIS

RM0CHKSUM
LIS

RM0FWASET
LIS

RM0EXTEND
LIS

RM0IFISI
LIS

RM0JOURNL
LIS

RM0FSETI
LIS

RM0FSET
LIS

RM0COMCLN
LIS

RM0DUMMY
LIS

RM0FILFNC
LIS

0319 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RMØRECLCK
LIS

RMØRSET
LIS

RMØSCAN
LIS

RMØPRFLNM
LIS

RMØRCLCK2
LIS

RMØRABCHK
LIS

RMØRELEAS
LIS

RMØNAMSTR
LIS